UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO
FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA, INFORMÁTICA Y MECÁNICA
ESCUELA PROFESIONAL DE INGENIERÍA INFORMÁTICA Y DE SISTEMAS

TESIS

INFLUENCIA DE LA CALIDAD DE IMAGEN EN EL RENDIMIENTO DE LAS REDES NEURONALES CONVOLUCIONALES APLICADAS A LA ESTIMACIÓN DEL PESO DEL CUY

Presentado por:
Br. WILLY ALDAIR CRUZ BEJAR

Para optar al título profesional de
INGENIERO INFORMÁTICO Y DE SISTEMAS

Asesor:
Dra. YESHICA ISELA ORMEÑO AYALA

CUSCO - PERÚ
2024

INFORME DE ORIGINALIDAD

(Aprobado por Resolución Nro.CU-303-2020-UNSAAC)

El que suscribe, **Asesor** del trabajo de investigación/tesis titulada: Influencia de la calidad de imagen en el rendimiento de las redes neuronales convolucionales aplicadas a la estimación del peso del cuy

presentado por: Willy Aldair Cruz Bejar con DNI Nro.: 73358871 presentado por: ..... con DNI Nro.: ..... para optar el título profesional/grado académico de Ingeniero Informático y de Sistemas

Informo que el trabajo de investigación ha sido sometido a revisión por 02 veces, mediante el Software Antiplagio, conforme al Art. 6° del ***Reglamento para Uso de Sistema Antiplagio de la UNSAAC*** y de la evaluación de originalidad se tiene un porcentaje de 6 **%.**

Evaluación y acciones del reporte de coincidencia para trabajos de investigación conducentes a grado académico o título profesional, tesis

Porcentaje	Evaluación y Acciones	Marque con una (X)
Del 1 al 10%	No se considera plagio.	X
Del 11 al 30 %	Devolver al usuario para las correcciones.	
Mayor a 31%	El responsable de la revisión del documento emite un informe al inmediato jerárquico, quien a su vez eleva el informe a la autoridad académica para que tome las acciones correspondientes. Sin perjuicio de las sanciones administrativas que correspondan de acuerdo a Ley.	

Por tanto, en mi condición de asesor, firmo el presente informe en señal de conformidad y **adjunto** la primera página del reporte del Sistema Antiplagio.

Cusco, 10 de octubre de 2024

.....
Firma

Post firma: Yeshica Isela Ormeño Ayala.

Nro. de DNI: 25002834

ORCID del Asesor: 0000-0002-5497-6928

Se adjunta:

1. Reporte generado por el Sistema Antiplagio.
2. Enlace del Reporte Generado por el Sistema Antiplagio: **oid:** 27259:390914227

NOMBRE DEL TRABAJO
Tesis_Willy_Aldair_Cruz_Bejar.pdf

AUTOR
Willy Aldair Cruz Bejar

RECUENTO DE PALABRAS
34715 Words

RECUENTO DE CARACTERES
185270 Characters

RECUENTO DE PÁGINAS
157 Pages

TAMAÑO DEL ARCHIVO
20.6MB

FECHA DE ENTREGA
Oct 10, 2024 9:50 PM GMT-5

FECHA DEL INFORME
Oct 10, 2024 9:52 PM GMT-5

6% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 5% Base de datos de Internet
- 1% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 4% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Material citado
- Coincidencia baja (menos de 12 palabras)

Agradecimientos

Quisiera expresar mi sincero agradecimiento a todas las personas que contribuyeron de manera significativa a la realización de esta tesis. En primer lugar, quiero agradecer especialmente a mis amados padres Willy y Marieta por su dedicación, amor y comprensión, a mi adorada hermana Maeba por su cariño y admiración incondicional, así como al resto de mi familia, colegas y amigos por su constante apoyo emocional y motivación.

Además, quiero expresar mi reconocimiento a mi alma mater, la Universidad Nacional de San Antonio Abad del Cusco, y en particular a los docentes de la escuela profesional de Ingeniería Informática y de Sistemas, quienes desempeñaron un papel crucial a lo largo de mi desarrollo profesional.

Mi sincero agradecimiento se extiende a mi asesora, la Dra. Yeshica Ormeño, cuya orientación experta y sabios consejos fueron fundamentales en cada fase de este proyecto.

Quiero concluir destacando que este logro no hubiera sido posible sin la dedicación y contribución de cada persona mencionada. Gracias a todos por ser parte de este viaje académico.

Resumen

La presente investigación aborda la influencia de la calidad de la imagen en el rendimiento de las Redes Neuronales Convolucionales (CNN) entrenadas para estimar el peso del cuy. Se simulan variaciones de calidad mediante cambios en la iluminación y la aplicación de siete distorsiones: desenfoque gaussiano, desenfoque de movimiento, ruido gaussiano, brillo, contraste, compresión JPEG y resolución. Los modelos ResNet50, DenseNet121 y Xception fueron evaluados en términos de error absoluto medio (MAE) y porcentaje de error absoluto medio (MAPE) utilizando un conjunto de datos de imágenes de cuyes de la región de Cusco. Los resultados muestran que las redes son sensibles a distorsiones como el ruido gaussiano y el contraste, afectando su capacidad para detectar el contorno, color y área del animal, mientras que el brillo y la resolución tienen un impacto mínimo. Entre los modelos, DenseNet121 se destaca como el más resistente a las distorsiones. Este estudio proporciona análisis fundamentales para mejorar la precisión y fiabilidad de las estimaciones de peso en cuyes, contribuyendo al avance de las aplicaciones de visión por computadora en la producción animal.

Palabras clave: redes neuronales convolucionales, calidad de imagen, estimación del peso animal, cuyes, distorsiones de imagen, visión por computadora

Abstract

This research addresses the influence of image quality on the performance of Convolutional Neural Networks (CNNs) trained to estimate the weight of guinea pigs. Quality variations are simulated by altering lighting conditions and applying seven distortions: Gaussian blur, motion blur, Gaussian noise, brightness, contrast, JPEG compression, and resolution. The ResNet50, DenseNet121, and Xception models were evaluated in terms of Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) using a dataset of guinea pig images from the Cusco region. The results show that the networks are sensitive to distortions such as Gaussian noise and contrast, affecting their ability to detect the outline, color, and area of the animal, while brightness and resolution have a minimal impact. Among the models, DenseNet121 stands out as the most robust against the studied distortions. This study provides fundamental analyses to improve the accuracy and reliability of weight estimations in guinea pigs, contributing to the advancement of computer vision applications in animal production.

Keywords: convolutional neural networks, image quality, animal weight estimation, guinea pigs, image distortions, computer vision

Índice General

Índice de Cuadros

Índice de Figuras

Índice de Código Fuente

Abreviaturas

ANN Red Neuronal Artificial

CNN Red Neuronal Convolucional

COCO Common Objects in Context

DCT Transformada Discreta del Coseno

DL Aprendizaje Profundo

DNN Red Neuronal Profunda

FPN Feature Pyramid Network

GAP Global Average Pooling

IA Inteligencia Artificial

ISBI Simposio Internacional sobre Imágenes Biomédicas

ISO Organización Internacional de Estandarización

JPEG Joint Photographic Experts Group

LSTM Long Short-Term Memory

MSE Error Cuadrático Medio

MAE Error Absoluto Medio

MAPE Error Absoluto Porcentual Medio

mAP Precisión Media Promedio

ML Aprendizaje automático

PPP Píxeles por pulgada

RAS Recirculating Aquaculture System

ReLU Rectified Linear Unit

RGB Rojo Verde Azul

YOLO You Only Look Once

VRAM Memoria de Acceso Aleatorio de Video

GPU Unidad de Procesamiento Gráfico

Introducción

La estimación del peso corporal de los animales ha sido objeto de extensa investigación, abordada desde diversas perspectivas que han evolucionado con el tiempo. Inicialmente, se emplearon análisis de imágenes junto con regresión lineal múltiple, utilizando medidas morfométricas de los animales, como se describe en el trabajo de Kashiha *et al.* (2014). Posteriormente, se avanzó hacia la utilización de modelos basados en Redes Neuronales Artificiales (ANN), aprovechando su capacidad para la regresión lineal, como se menciona en el estudio de Wang *et al.* (2008). En la actualidad, la tendencia se orienta hacia el uso de Redes Neuronales Convolucionales (CNN) para este propósito, utilizando una variedad de tipos de imágenes, como las 2D (Jun *et al.*, 2018; Zapata-Ttito, 2022), 3D (Hansen *et al.*, 2018) e incluso imágenes de profundidad (Meckbach *et al.*, 2021).

Sin embargo, se presta escasa atención a la calidad de los datos, un factor crucial en el desempeño de las CNN entrenadas para la regresión. Los datos de imágenes tienen una fuerte influencia en el funcionamiento adecuado de estos modelos. Esto se debe a que los entornos de aplicación de los modelos entrenados no son entornos controlados y muchas veces presentan condiciones adversas, como en granjas y criaderos, donde la variabilidad en la iluminación y las limitaciones de la tecnología de las cámaras pueden afectar considerablemente la calidad de las imágenes y, en consecuencia, el rendimiento del modelo.

En relación con la calidad de la imagen, Wueller y Kejser (2016) defienden que esta se caracteriza por la fidelidad en la reproducción de tonos, colores, detalles y geometría del original, así como por la presencia de artefactos (como ruido, píxeles defectuosos o aliasing). Basado en esta definición, el presente estudio tiene como objetivo examinar el impacto de la modificación de estos aspectos de la imagen en el rendimiento de modelos de CNN entrenados para la estimación del peso animal, específicamente el peso de los cuyes.

Se espera que este estudio proporcione a futuros investigadores pautas para el diseño de sistemas de visión artificial prácticos, considerando el impacto de la calidad de la imagen y el entorno visual. De esta forma, se podrán desarrollar modelos de Redes Neuronales Convolucionales más robustos, capaces de manejar imágenes de baja calidad y funcionales bajo condiciones adversas.

Aspectos Generales

Problema de Investigación

Planteamiento del Problema

Actualmente, la inteligencia artificial desempeña un papel crucial en el apoyo a los productores ganaderos en la gestión de sus granjas, especialmente en la toma de decisiones críticas relacionadas con la producción. Esto ha dado origen al campo de la ganadería de precisión, donde se emplean ampliamente técnicas de Visión Artificial y Redes Neuronales Convolucionales (CNN) para recopilar información sobre la salud y el bienestar del ganado, como la evaluación de la condición corporal, el peso y el comportamiento (Qiao *et al.*, 2019). En este contexto, una de las principales necesidades es mejorar la precisión y la confiabilidad en la estimación del peso de los animales.

La estimación del peso en animales de granja, como vacas (Gjergji *et al.*, 2020), cerdos (Jun *et al.*, 2018; Buayai *et al.*, 2019; Meckbach *et al.*, 2021), ovejas (Sant'Ana *et al.*, 2021) y peces (Konovalov *et al.*, 2019; Fernandes *et al.*, 2020), ha sido objeto de diversas investigaciones utilizando modelos de CNN, logrando rendimientos y precisiones notables. Sin embargo, estos sistemas de visión artificial suelen entrenarse y evaluarse con imágenes de alta calidad. En la práctica, se enfrentan diversos escenarios donde no se puede asumir tal nivel de calidad (Dodge y Karam, 2016). Como resultado, el rendimiento de estos modelos en entornos reales no siempre alcanza el nivel óptimo observado en condiciones ideales.

Según Ranjan *et al.* (2023), los métodos de Aprendizaje automático (ML) basados en CNN requieren grandes cantidades de datos y la precisión del modelo depende en gran medida de la calidad de las imágenes de entrada. Si bien se han realizado estudios sobre los efectos de la calidad de imagen en el rendimiento de las CNN para tareas como clasificación (Dodge y Karam, 2016), reconstrucción de imágenes (Jeelani *et al.*, 2018), mapeo de hierbas (Hu *et al.*, 2021), y detección de enfermedades (Akkoca Gazioğlu y

Kamaşak, 2021), existe una carencia de investigación específica sobre cómo la calidad de la imagen afecta el rendimiento de las CNN en tareas de regresión. Hasta la fecha, no se ha explorado exhaustivamente cómo las variaciones en la calidad de las imágenes impactan la estimación del peso en animales, especialmente en cuyes. Por ello, esta investigación se propone analizar cómo diferentes niveles de calidad de imagen afectan el desempeño de modelos de CNN en la estimación del peso de cuyes.

Formulación del Problema

Problema General

- ¿Cuáles son los efectos de las variaciones en la calidad de imagen en el rendimiento de las redes neuronales convolucionales aplicadas a la estimación del peso del cuy?

Problemas Específicos

1. ¿Cuáles son las distorsiones de calidad de imagen que podrían influir en el rendimiento de redes neuronales convolucionales entrenadas para estimar el peso corporal de los cuyes?
2. ¿Qué dataset de imágenes de cuyes es necesario para analizar el efecto de las variaciones en la calidad de imagen en el rendimiento de modelos CNN?
3. ¿Qué modelos de CNN muestran el mejor rendimiento para estimar el peso del cuy utilizando imágenes con variaciones en la calidad?

Antecedentes

En los últimos años, se han realizado numerosas investigaciones sobre el desarrollo de modelos de estimación de peso animal utilizando tanto Redes Neuronales Convolucionales (CNN) como Redes Neuronales Profundas (DNN). Sin embargo, se ha reconocido que el aprendizaje automático asistido por estas redes está influenciado por la calidad de las imágenes utilizadas.

En este contexto, algunas investigaciones se han centrado en el análisis de los efectos de la calidad de la imagen en el rendimiento de las CNN entrenadas para diversas

tareas. A continuación, se presentan las investigaciones más destacadas en este ámbito:

- **"Understanding how image quality affects Deep Neural Networks"** Samuel Dodge, Lina Karam (Dodge y Karam, 2016)

  Esta investigación tiene como objetivo proveer una evaluación a gran escala de cuatro modelos de DNN de última generación entrenados para la clasificación de imágenes naturales bajo diferentes tipos y niveles de distorsiones de calidad de la imagen. Además, también busca determinar qué arquitecturas serían útiles para construir redes más invariantes a las distorsiones presentadas.

  La metodología experimental utilizada se describe de la siguiente forma: 1) Establecer las redes profundas, 2) Establecer el conjunto de datos, 3) Establecer los tipos y niveles de distorsión y, finalmente, 4) Realizar el análisis y contraste de los resultados.

  Los modelos considerados son: Caffe Reference, VGG-CNN-S, VGG-16 y GoogleNet.

  El conjunto de datos utilizado es ImageNet, que cuenta con 1,000 categorías de imágenes y con 1.2 millones de imágenes de entrenamiento. Para el subconjunto de validación, con el objetivo de ahorrar tiempo de computación, solo se consideraron 10,000 de las 50,000 imágenes disponibles, escogiendo 10 imágenes de cada una de las 1,000 categorías. Posteriormente, para cada imagen se generaron imágenes adicionales con distintos niveles de distorsión en la calidad.

  Las distorsiones aplicadas son: Compresión JPEG, JPEG 2000, ruido, desenfoque y contraste. Cada una de estas distorsiones fue aplicada y evaluada por separado.

  Los resultados más relevantes de la investigación son: todas las redes son muy sensibles al desenfoque, las imágenes con ruido muestran una disminución similar del rendimiento, las redes son sorprendentemente resistentes a las distorsiones de la compresión JPEG y JPEG 2000, y las redes también muestran resistencia a los cambios de contraste.

Una conclusión relevante es que el entrenamiento con imágenes de baja calidad debería mejorar los resultados de las pruebas en imágenes de baja calidad, aunque esto también podría dar lugar a una disminución del rendimiento en imágenes de alta calidad.

- **"Effects of image data quality on a convolutional neural network trained in-tank fish detection model for recirculating aquaculture systems"** Rakesh Ranjan, Kata Sharrer, Scott Tsukuda, Christopher Good (Ranjan *et al.*, 2023)

  El objetivo de este estudio es investigar los efectos de la calidad de los datos de las imágenes, el tamaño de los datos, las condiciones de las imágenes y las operaciones de preprocesamiento en la precisión del modelo CNN de detección individual de peces. Se definen dos objetivos específicos: 1) Desarrollo de una plataforma de detección subacuática (RASense1.0) para la adquisición de datos de imagen en un ambiente Recirculating Aquaculture System (RAS) (Sistema de Recirculación Acuícola) y 2) Investigar el efecto de la selección del sensor, la condición de la imagen, el tamaño del conjunto de datos y el ajuste de los hiperparámetros de la red en el rendimiento del modelo CNN.

  La metodología utilizada en este estudio es la siguiente: 1) Diseñar y construir la plataforma de detección subacuática (RASense1.0), 2) Instalación de la plataforma de detección, 3) Adquisición de datos, 4) Preprocesamiento de datos, 5) Selección del modelo de detección de peces, 6) Análisis de datos.

  La plataforma de detección subacuática se desarrolló con cuatro sensores de imagen comerciales y se instaló en un tanque RAS localizado en The Conservation Fund Freshwater Institute (Shepherdstown, WV, Estados Unidos).

  Los datos de imágenes se recogieron en condiciones de luz ambiental y con luz suplementaria. Luego se descargaron y se inspeccionaron manualmente para identificar y filtrar las imágenes idénticas.

  Para el modelo de detección, se adoptó un modelo de detección de objetos de una

etapa You Only Look Once (YOLO). YOLOv5 se utilizó para la detección de peces en el tanque. La unión de los datasets se dividió en una proporción de 70:20:10 para el entrenamiento, la validación y la prueba del modelo, respectivamente. Para el aumento de datos se probaron siete modificaciones (brillo [±25 %], exposición [±25 %], saturación [±50 %], ruido [±5 %], desenfoque [5 px], recorte [5 %] y mosaico). Por último, se adoptó y entrenó el modelo CNN de dos etapas Faster R-CNN con ayuda de una biblioteca de código abierto Detectron2 (Meta Research, California, EE.UU.) y se comparó su rendimiento con el modelo YOLOv5.

Las conclusiones de los autores son las siguientes: el modelo desarrollado de detección de peces en una sola etapa alcanzó una Precisión Media Promedio (mAP) y F1 score satisfactorias del 86,5 % y 0,8, respectivamente. Al aumentar el tamaño de los datasets y los valores de las épocas hasta 700 y 80, se observó un aumento logístico de la mAP y F1 score. La selección del sensor de imagen afectó considerablemente al rendimiento del modelo; sin embargo, las condiciones de luz no mostraron efectos sustanciales.

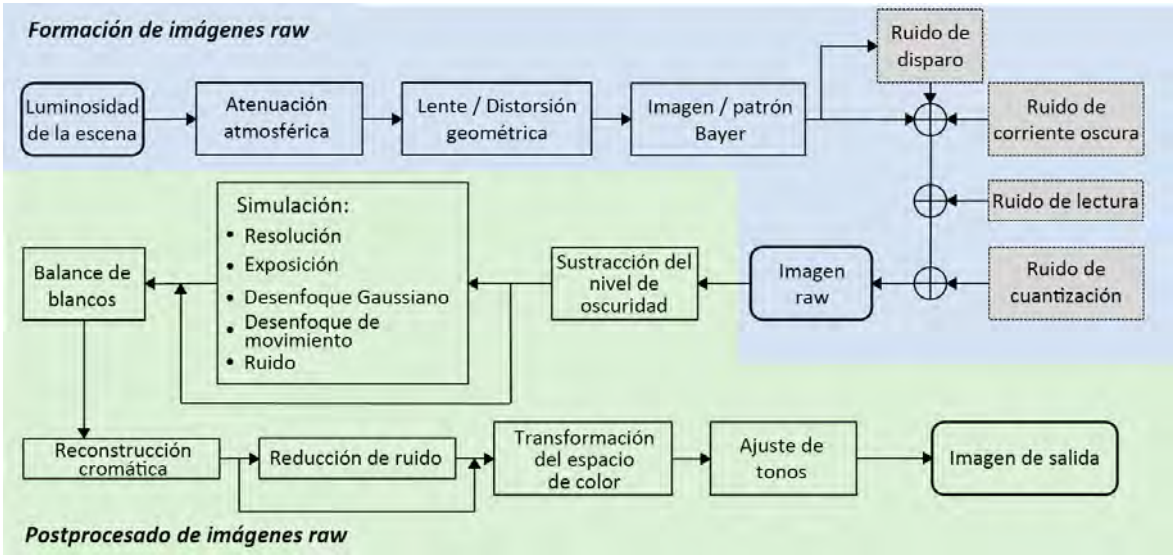
- **"Influence of image quality and light consistency on the performance of Convolutional Neural Networks for weed mapping"** Chengsong Hu, Bishwa B. Sapkota, J. Alex Thomasson, Muthukumar V. Bagavathiannan (Hu *et al.*, 2021)

  En esta investigación se pretende comprender mejor la influencia de la calidad de la imagen y la consistencia de la luz en el rendimiento de las CNN para el mapeo de hierbas mediante la simulación del proceso de formación de imágenes (Figura 1). Dentro de este proceso se realiza la introducción de varias degradaciones de la calidad de la imagen (reducción de resolución, sobreexposición, desenfoque gaussiano, desenfoque de movimiento y ruido).

  La metodología usada en este trabajo es la siguiente: 1) Recolección de datos, 2) Caracterización de la cámara, 3) Simulación, 4) Anotación de las imágenes, 5) Entrenamiento y evaluación de redes neuronales, 6) Elección de métricas de evaluación.

Figura 1

Proceso de formación de imágenes raw y postprocesado



Nota. Adaptado de Influence of Image Quality and Light Consistency on the Performance of Convolutional Neural Networks for Weed Mapping (p. 3), por Hu et al., 2021, Remote Sensing, 13(11):2140. CC BY

Los tres conjuntos de imágenes recolectados se denominaron S_{noon}, S_{sunset} y S_{cloudy}. Se consideraron a las imágenes raw sin degradaciones como "verdad fundamental". Un total de 1485 imágenes fueron anotadas para DS_{noon}, 500 para DS_{sunset} y 500 para DS_{cloudy}.

En el entrenamiento, los modelos de CNN fueron entrenados y probados en tres situaciones: 1) Las imágenes de entrenamiento y de prueba tienen la misma calidad, 2) Las imágenes de entrenamiento y de prueba tienen inconsistencia de calidad, 3) Las imágenes de entrenamiento y de prueba tienen inconsistencia de luz. Adoptaron tres modelos para evaluar su rendimiento: un modelo Faster R-CNN estándar con ResNet50 + Feature Pyramid Network (FPN) backbone para la detección de objetos, la misma configuración anterior para el modelo Mask R-CNN para la segmentación de instancias y, por último, el modelo Deeplab-v3 se utilizó con el backbone ResNet50 para la segmentación semántica.

Las conclusiones de los autores fueron que lograron simular las degradaciones de imagen más comunes observadas en las aplicaciones de mapeo de hierbas a través

del proceso de formación de imágenes. Además, exploraron la influencia de estas degradaciones en el rendimiento de los tres modelos CNN ampliamente utilizados, Faster R-CNN, Mask R-CNN y Deeplab-v3, para la detección de objetos, la segmentación de instancias y la segmentación semántica, respectivamente. La simulación de la degradación de las imágenes se basó en las imágenes raw, que inevitablemente contienen ruido y desenfoque. Aunque intentaron reducir al mínimo estas degradaciones, no fue posible eliminarlas por completo. Por lo tanto, concluyen que aún se desconoce cuál es el mejor rendimiento de las CNN que puede lograrse con imágenes perfectas.

- **"Effects of objects and image quality on melanoma classification using deep neural networks"** Bilge S. Akkoca Gazioglu, Mustafa E. Kamasak (Akkoca Gazioğlu y Kamaşak, 2021)

  El objetivo de este trabajo es investigar los efectos de los objetos externos (regla, pelo) y la calidad de la imagen (desenfoque, ruido, contraste) en la clasificación del melanoma utilizando modelos de Redes Neuronales Convolucionales de uso común: ResNet50, DenseNet121, VGG16 y AlexNet.

  La metodología utilizada sigue los siguientes pasos: 1) Recopilación del conjunto de datos, 2) Aumento de datos, 3) Pruebas con imágenes degradadas y 4) Selección de modelos de DNN.

  Para el conjunto de datos se combinaron los datasets del Simposio Internacional sobre Imágenes Biomédicas (ISBI) lanzados en 2017, 2018 y 2019, reduciendo las clases a solo dos: benigno y melanoma. Las particiones se hicieron en una proporción aproximada del 80-20 % para entrenamiento y prueba, respectivamente. Para analizar los efectos de los objetos externos a la imagen, se formaron tres subconjuntos de prueba: $Test_{Ruler}$, $Test_{Hair}$ y $Test_{None}$.

  Para el aumento de datos se utilizaron la rotación y la inversión; además, las imágenes se giraron 45° y se voltearon en horizontal, vertical y en ambos ejes.

Para realizar las pruebas con las imágenes degradadas se usaron diferentes niveles de degradación como desenfoque, ruido y contraste. Se usaron las imágenes del conjunto de datos $Test_{None}$ para ambas clases.

Los modelos seleccionados fueron: ResNet, DenseNet, AlexNet y VGGNet. A cada uno de estos modelos se le removió la última capa totalmente conectada y se le agregó el clasificador personalizado diseñado por los autores.

Los resultados más llamativos de la investigación son que en los cambios de contraste, las imágenes de melanoma son más robustas que las de lesiones benignas. Con la ampliación del tamaño del filtro utilizado en el desenfoque, las imágenes se deterioran considerablemente. Se observa una rápida disminución con el modelo AlexNet y VGGNet. Con respecto a los objetos, mencionan que los objetos adicionales en las imágenes suelen disminuir el rendimiento del reconocimiento. En la prueba de la regla, DenseNet, con una precisión media del 86 %, obtuvo mejores resultados que otros modelos CNN. Además, ResNet se acercó más a DenseNet con una precisión del 85,23 %. Por consiguiente, considerando 6 conjuntos de pruebas, DenseNet tuvo una mejor precisión de clasificación en lo que respecta a los problemas de degradación de la imagen.

- **“Aplicación de visión artificial en la estimación del peso corporal del cuy”**
  Zapata Ttito, Abel Gabriel (Zapata-Ttito, 2022)

  En este trabajo se busca desarrollar un sistema de visión artificial aplicado a la estimación del peso corporal del cuy utilizando una arquitectura de red neuronal convolucional.

  La metodología para el desarrollo del modelo que el autor plantea es la siguiente: 1) Revisión bibliográfica, 2) Recolección de datos, 3) Procesamiento de imágenes, 4) Construcción del dataset, 5) Desarrollo de la arquitectura CNN, 6) Análisis de resultados.

  En la etapa de revisión bibliográfica, el autor hace referencia a diversos trabajos de

estimación del peso vivo de otros animales, tales como cerdos, vacas y peces.

La recolección de datos comprende la captura de 225 imágenes de cuyes y la obtención de sus pesos usando una balanza digital en granjas de crianza de la región Cusco. Para la segmentación del cuy del fondo de la imagen, se utilizó el software Labelbox, generando así las máscaras para cada imagen. Finalmente, antes de realizar el aumento de datos, se procedió a dividir el dataset para el entrenamiento (80 %) y la validación (20 %). Para el aumento de datos, por cada imagen original se obtuvieron tres imágenes adicionales.

Para el desarrollo de la arquitectura de CNN, se optó por el modelo Mask R-CNN, diseñado para la segmentación de instancias, ya que permite generar con precisión una máscara del objeto a identificar. Utilizando dos redes troncales (ResNet50 y ResNet101) y pesos preentrenados de los datasets Common Objects in Context (COCO) e ImageNet, se diseñaron dos modelos: el modelo A (peso y máscara en la misma rama) y el modelo B (una rama solo para el peso).

Los resultados más destacados respecto a la estimación del peso son los siguientes:

En el modelo A, el resultado óptimo obtenido pertenece al modelo ResNet50A, con una correlación de 72.38 % y un valor de Error Absoluto Porcentual Medio (MAPE) de 14.20 %, todo esto evaluado con el conjunto de validación. En el modelo B, el mejor resultado que se obtuvo fue una correlación de 74.44 % y un MAPE de 14.17 % usando la red troncal ResNet50 y los pesos de COCO, considerándose esto solo como una leve mejora con respecto al modelo ResNet50A. Luego del refinamiento, el mejor resultado promedio que se obtuvo fue de 79.45 % y 77.20 % con el modelo ResNet50B. Sin embargo, el mejor resultado de todos fue una correlación de 80.09 % usando el modelo ResNet50A, además de un MAPE de 12.41 %.

En resumen, esta revisión bibliográfica ha permitido identificar algunas de las técnicas y enfoques más relevantes para investigar el impacto de la calidad de la imagen en las CNN. Sin embargo, se ha destacado una brecha en la literatura, ya que la mayoría de los

estudios se centran en modelos entrenados para tareas de clasificación y detección, mientras que aún no se ha realizado ninguna investigación que aborde específicamente el rendimiento de las CNN en tareas de regresión. A partir de la información recopilada en la literatura revisada, se consideran las metodologías, los tipos de distorsiones de calidad de imagen y las arquitecturas evaluados para plantear un experimento que aborde esta brecha y contribuya a llenar este vacío en la investigación.

Justificación

Entender el impacto de la calidad de la imagen en el rendimiento de las arquitecturas de Redes Neuronales Convolucionales (CNN) es fundamental para desarrollar nuevas y mejores arquitecturas y optimizar las existentes. En este sentido, se han realizado estudios previos sobre el impacto de la calidad de imagen en el rendimiento de las CNN entrenadas para diversas tareas, como clasificación de imágenes (Dodge y Karam, 2016), reconstrucción de imágenes de resonancia magnética (Jeelani *et al.*, 2018), mapeo de hierbas (Hu *et al.*, 2021), clasificación de melanoma (Akkoca Gazioğlu y Kamaşak, 2021), y detección de peces (Ranjan *et al.*, 2023). Generalmente, estos trabajos se enfocan en tareas de clasificación y detección, sin abordar el estudio de modelos entrenados para la regresión.

Además, según Dodge y Karam (2016), con la proliferación de aplicaciones de visión por ordenador en teléfonos móviles, podría ser necesario relajar el requisito de imágenes de alta calidad, considerando las limitaciones de los dispositivos móviles y la tendencia de las tecnologías a trasladarse a estos dispositivos debido a su versatilidad y masividad. Por lo tanto, al ser utilizados como fuente de captura de imágenes mediante sus cámaras integradas, la calidad de imagen adquiere una relevancia aún mayor.

La calidad de imagen es un aspecto práctico y relevante que a menudo se pasa por alto en el diseño de sistemas de visión artificial, aunque es fundamental para alcanzar un rendimiento óptimo en las CNN. Según Ranjan *et al.* (2023), la precisión de un modelo depende de la calidad de la imagen de entrada. Dado su carácter crucial, es esencial investigar exhaustivamente su influencia para entender cómo se comportan las redes en

diversas condiciones de calidad. Este enfoque permitirá evaluar qué atributos de la imagen afectan más el rendimiento de los modelos. Al analizar estos efectos, se podrán obtener conclusiones significativas y determinar cuáles arquitecturas son más sensibles a niveles bajos de calidad de imagen y cuáles serían útiles para desarrollar nuevos modelos más resistentes a diferentes niveles de calidad. Esto sería de gran interés para los investigadores, proporcionándoles orientación en el desarrollo de sistemas de visión prácticos y adaptados a condiciones no ideales, capaces de manejar imágenes de baja calidad. Establecer un umbral de calidad en los modelos de CNN podría servir como referencia para la selección de parámetros de cámara en futuras aplicaciones (Hu *et al.*, 2021).

En el contexto social, este estudio es significativo al enfocarse en el cuy, dado su papel crucial tanto en la alimentación como en la economía de la región Cusco. La cría de cuyes es una práctica común en las granjas locales, y según el Ministerio de Desarrollo Agrario y Riego (2023), Cusco es la segunda región productora de cuyes a nivel nacional. Además, como menciona Zapata-Ttito (2022), los productores y las empresas ganaderas se benefician al mejorar y dotar de mayor precisión a las herramientas no intrusivas que utilizan para monitorear el peso de sus animales, lo que contribuye a una gestión más eficaz de las granjas y, en consecuencia, a una crianza mejorada.

Objetivos

Objetivo General

Determinar los efectos de las variaciones en la calidad de imagen sobre el rendimiento de las redes neuronales convolucionales en la estimación del peso del cuy.

Objetivos Específicos

1. Identificar las distorsiones de calidad de imagen que podrían influir en el rendimiento de redes neuronales convolucionales entrenadas para estimar el peso corporal de los cuyes.
2. Desarrollar un dataset de imágenes de cuyes que incluya variaciones en la calidad

de imagen.

3. Determinar cuál modelo demuestra mejor rendimiento con imágenes que presentan variaciones en la calidad.

Alcance y Limitaciones

Para establecer los límites y enfoques específicos de esta investigación, se detallan a continuación las diversas delimitaciones que enmarcan este trabajo.

Alcance

- La investigación se centra en evaluar diferentes niveles de calidad de imagen y su impacto en la precisión de las predicciones de las CNN. Los diferentes niveles de calidad son simulados a partir de la introducción de distorsiones en las imágenes originales. Así mismo cada distorsión y nivel de distorsión son introducidos por separado.
- El estudio abarca un período de 16 semanas, con 20 horas de trabajo por semana. Este marco temporal ha sido elegido para permitir una recopilación suficiente de imágenes y la realización de múltiples pruebas con las redes neuronales.
- La investigación se lleva a cabo utilizando imágenes de cuyes provenientes de criaderos de la región Cusco, debido a la alta producción de cuyes y la relevancia económica de este animal en dicha área.
- La población objeto del estudio está constituida por imágenes de cuyes de diferentes edades y pesos, obtenidas en condiciones controladas de iluminación y entorno para minimizar variables externas. No se incluyeron cuyes que presenten condiciones anómalas de salud que puedan afectar su peso de manera significativa.
- Este estudio se enfoca en un conjunto específico de arquitecturas de redes neuronales (como ResNet50, DenseNet121 y Xception). La elección de estos modelos puede no abarcar todas las posibles técnicas de aprendizaje profundo que podrían ser aplicables al problema. Otras arquitecturas y enfoques podrían

proporcionar diferentes perspectivas y resultados.

Limitaciones

- La creación de un dataset amplio resulta inviable debido a la limitada disponibilidad de ejemplares de cuyes para la captura de imágenes en cada granja. Esta restricción se origina por diversos factores, como la precaución de algunos productores en cuanto al traslado de cuyes en estado de gestación, cuyes reproductores o crías.
- El acceso a las granjas de cuyes se ve limitado por la disponibilidad de tiempo de los productores ganaderos de cuyes.
- La diversidad en términos de edad, tamaño y condiciones de salud de los cuyes puede no estar completamente cubierta, lo que podría limitar la aplicabilidad de los resultados a poblaciones más amplias de cuyes.
- La cámara empleada para la captura de imágenes del dispositivo móvil (Xiaomi Redmi 5 Plus), que aunque cuenta con especificaciones aceptables, no ofrece la misma resolución, rango dinámico y control de exposición que cámaras profesionales. Esto puede afectar la precisión de la segmentación y el detalle capturado en las imágenes, impactando la calidad del dataset.
- El entrenamiento de redes neuronales convolucionales requiere de una capacidad computacional significativa. Algunas limitaciones en los recursos disponibles, como una Unidad de Procesamiento Gráfico (GPU) de alto rendimiento y tiempo de procesamiento, restringen el tamaño de los modelos que se pueden entrenar y el número de experimentos que se pueden realizar.
- El procesamiento y entrenamiento de los modelos se realizó en una máquina virtual de Google Colaboratory utilizando únicamente el plan gratuito. Esta versión tiene restricciones en la memoria RAM, almacenamiento y disponibilidad de la GPU, lo cual limita la ejecución de redes neuronales más complejas y la manipulación de datasets de mayor tamaño, afectando la velocidad de entrenamiento y el número de experimentos posibles.

Metodología

Tipo de Investigación

En esta investigación se utiliza un enfoque cuantitativo, debido a que se busca medir y estimar magnitudes como el rendimiento de los modelos CNN por medio de la estadística (Hernández-Sampieri *et al.*, 2014).

Según Hernández-Sampieri *et al.* (2014), "Un experimento es una situación de control en la cual se manipulan, de manera intencional, una o más variables independientes (causas) para analizar las consecuencias de tal manipulación sobre una o más variables dependientes (efectos)" (p.130). Por lo tanto, la metodología de este trabajo se considera de tipo experimental, ya que se busca medir la variación de la precisión de los modelos CNN (variable dependiente) en base a las manipulaciones de la calidad de la imagen de entrada (variable independiente).

Metodología Propuesta

La metodología propuesta para este proyecto de investigación se basa en las investigaciones mencionadas en los antecedentes que siguen un proceso metodológico similar. Por tanto, se propone seguir una serie de pasos similares.

El desarrollo metodológico, tal como se ilustra en la Figura 2, sigue los siguientes pasos:

1. **Captura de datos**

   El proceso de captura de datos involucra la toma de imágenes del cuy con una cámara fotográfica bajo dos condiciones de iluminación: luz ambiental y luz suplementaria, y la medición de su peso con una balanza. La preparación del entorno (ajuste de la iluminación, posición de la cámara y balanza) también se considera un aspecto importante.

2. **Selección de factores y distorsiones de calidad de la imagen**

   Esta etapa implica una revisión bibliográfica basada en el análisis de artículos

científicos relacionados con la evaluación de los efectos de la calidad de imagen en las CNN y evaluación de la calidad de imagen. Para ello, se recopilan estos recursos en bibliotecas digitales.

3. **Procesamiento de imágenes**

   El procesamiento de imágenes consta de dos etapas:

   a) **Selección de imágenes**

   En esta fase inicial, se eligen imágenes que capturen de manera adecuada el cuerpo completo del cuy. Luego, se eliminan las imágenes similares.

   b) **Aplicación de las distorsiones de calidad y aumento de datos**

   En esta fase, se aplican las distorsiones de calidad previamente seleccionadas al conjunto de imágenes de prueba. Además, se amplía el conjunto de datos mediante la rotación, sin considerar otros métodos de aumento de datos, para no interferir con la aplicación de las distorsiones de calidad.

4. **Construcción de datasets**

   La creación de los conjuntos de datos se lleva a cabo agrupando los datos (imágenes y pesos) que pertenecen a una misma categoría. Primero, se genera el conjunto de datos principal. A partir de este conjunto principal, se forman otros conjuntos con características similares (iluminación, tipo de distorsión). Posteriormente, cada uno de estos conjuntos se divide en proporciones de 80:10:10 para ser utilizados en entrenamiento, validación y prueba, respectivamente.

5. **Selección, implementación y entrenamiento de modelos de CNN**

   a) **Selección de los modelos**

   La selección se lleva a cabo mediante una revisión bibliográfica basada en el análisis de artículos científicos centrados en regresión y clasificación utilizando diversas técnicas de visión artificial y comparativas de modelos

y evaluación de rendimiento.

b) **Implementación de los modelos**

La implementación se realiza accediendo a repositorios que contienen el código fuente del modelo elegido. En situaciones donde dicho código no esté disponible, se procede a seguir las indicaciones proporcionadas en la investigación para realizar la implementación. Si es imprescindible, se opta por implementar un modelo con una arquitectura similar.

c) **Entrenamiento de los modelos**

La fase de entrenamiento involucra el proceso de entrenar y optimizar los hiperparámetros de los modelos elegidos utilizando los distintos conjuntos de datos generados en las etapas anteriores.

6. **Evaluación y comparativa de rendimiento**

En esta etapa se definen dos tareas importantes:

a) **Construcción de gráficos y tablas**

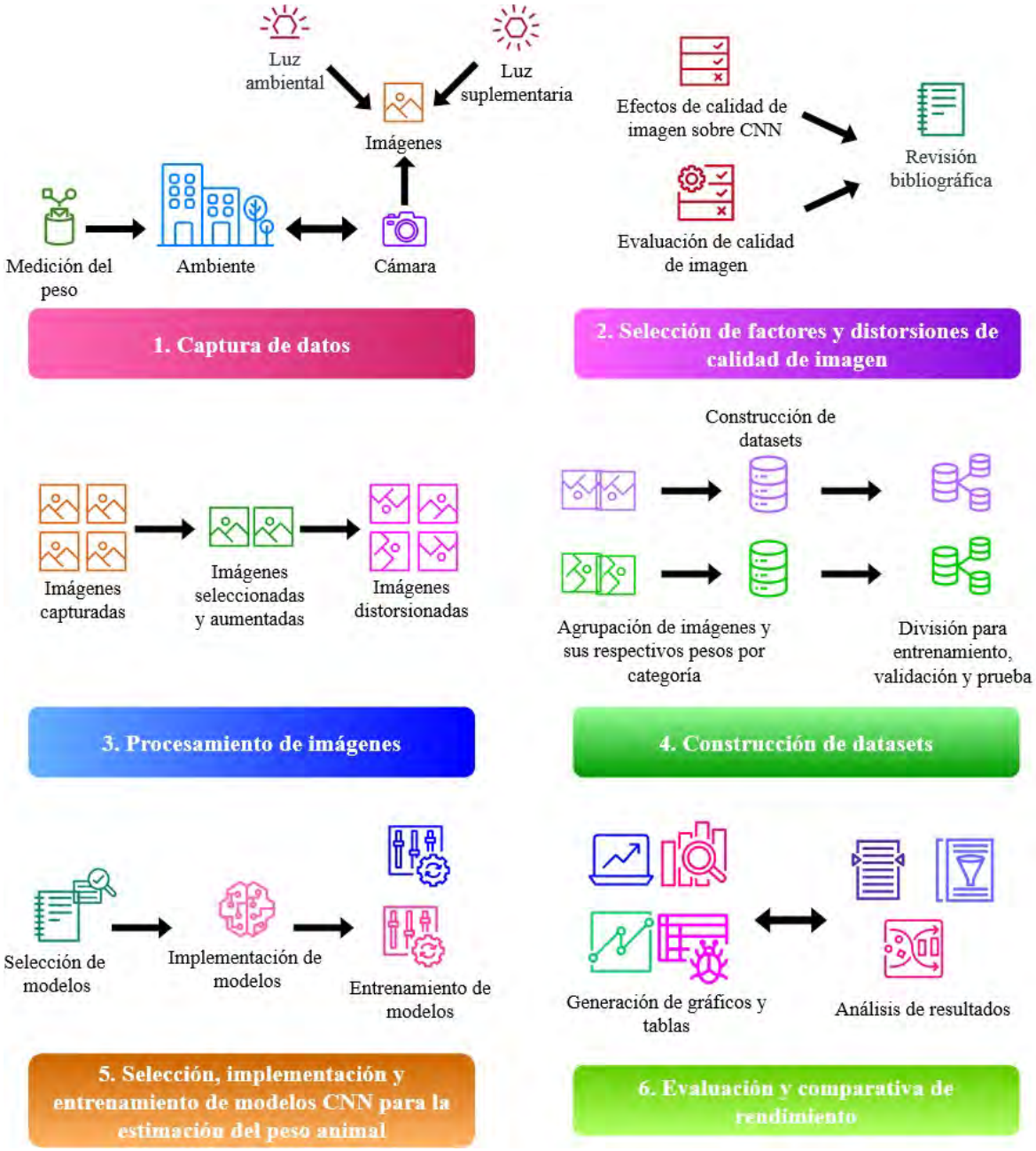
En esta etapa se generan los gráficos y tablas comparativas en base al rendimiento obtenido por los modelos en su entrenamiento y prueba.

b) **Análisis de resultados**

En la etapa de análisis, el objetivo es interpretar los resultados, examinando las posibles razones y fenómenos que hayan conducido a los modelos a obtener cierto nivel de rendimiento. Este análisis se realiza en base a los gráficos y tablas resultantes.

Figura 2

Metodología propuesta



En resumen, la metodología propuesta para este proyecto de investigación proporciona un enfoque integral y basado en datos para investigar los efectos de las variaciones de la calidad de las imágenes en el rendimiento de los modelos de CNN para la estimación del peso del cuy. Esta metodología se fundamenta en las técnicas de aprendizaje automático y visión artificial, y se espera que proporcione resultados significativos y aplicables a este campo de estudio.

Resultados Esperados

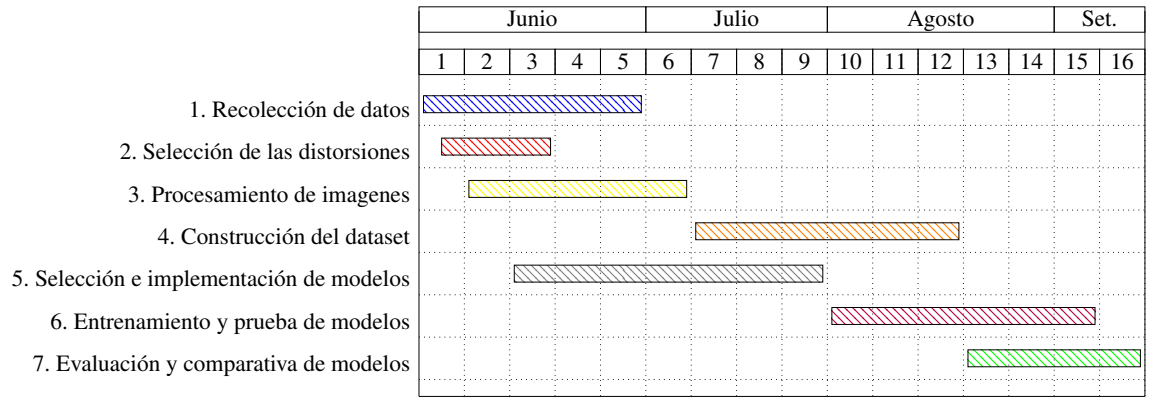
Los resultados esperados en términos de entregables concretos son los siguientes:

1. Un dataset de imágenes y pesos de cuyes bajo dos tipos de iluminación de todos los cuyes recolectados. Este dataset incluirá imágenes tomadas en condiciones de luz ambiental y luz suplementaria.
2. Tres modelos de CNN entrenados para la estimación del peso del cuy. Los modelos serán entrenados utilizando técnicas de preprocesamiento y aumento de datos para mejorar la precisión y robustez de las predicciones.
3. El análisis de resultados y comparativa del rendimiento de los modelos bajo las dos condiciones de iluminación y bajo los datasets de prueba con imágenes distorsionadas. Este análisis incluirá métricas como el error medio absoluto (MAE) y el error absoluto medio porcentual (MAPE), comparando cómo cada modelo se desempeña en diferentes escenarios de iluminación y bajo la influencia de las distorsiones seleccionadas.

Cronograma de Actividades

El cronograma de actividades del presente trabajo que se muestra en la Figura 3 se ajusta a 16 semanas y 20 horas de trabajo por semana por persona.

Figura 3
Diagrama de Gantt del cronograma de actividades.



Recursos y Presupuesto

En el Cuadro 1 se presenta el listado de recursos y el presupuesto de la investigación.

Cuadro 1

Presupuesto de la investigación

Categoría	Recurso	Descripción	Unidad	Cantidad	Precio Unitario	Subtotal
Recursos humanos	Investigador principal	Principal encargado de la investigación	-	1	S/ 0.00	S/ 0.00
	Expertos y asesores	Honorarios por consultas y asesoramiento especializado durante el desarrollo de la investigación.	-	1	S/ 0.00	S/ 0.00
	Acceso a internet	Para la búsqueda bibliográfica y acceso a recursos digitales	mes	9	S/ 80.00	S/ 720.00
	Impresiones	Para la impresión de borradores del documento	unidad	9	S/ 13.30	S/ 119.70
Materiales y Equipos	Papel Bond A4	Para la impresión de borradores del documento	paquete	4	S/ 15.50	S/ 62.00
	Equipo móvil	Trípode	unidad	1	S/ 89.90	S/ 89.90
	Equipo móvil	Fuente de iluminación	unidad	1	S/ 150.00	S/ 150.00
	Equipo móvil	Balanza digital	unidad	1	S/ 0.00	S/ 0.00
	Equipo móvil	Laptop personal para la documentación y desarrollo del proyecto	unidad	1	S/ 0.00	S/ 0.00
	Equipo móvil	Cámara (teléfono inteligente)	unidad	1	S/ 0.00	S/ 0.00
	Vehículo	Auto personal para traslados a locaciones cercanas	unidad	1	S/ 0.00	S/ 0.00
Acceso a bases de datos de literatura científica	Suscripciones y acceso a base de datos	Suscripciones a revistas científicas y acceso a bases de datos en línea para la búsqueda y revisión de literatura relevante.	unidad	10	S/ 68.90	S/ 68.90
	Gastos de membresía de bibliotecas virtuales	Gastos de membresía o acceso a bibliotecas digitales y recursos académicos	unidad	1	S/ 22.70	S/ 45.40
Gastos de viaje y viáticos	Transporte móvil	Gasolina	galón	2	S/ 12.00	S/ 96.00
	Transporte móvil	Pasajes interprovinciales	unidad	8	S/ 1.00	S/ 20.00
	Transporte móvil	Pasajes urbanos	unidad	20	S/ 11.00	S/ 165.00
	Manutención	Refrigerios para el trabajo de campo	unidad	15	S/ 45.00	S/ 225.00
	Alojamiento	Alojamiento en locaciones de toma de muestras	unidad	5	S/ 337.80	S/ 675.60
Software y licencias	Herramientas para Latex	Membresía Overleaf para la redacción del documento	año	2	S/ 0.00	S/ 0.00
	Software especializado	Anaconda, Python, Tensorflow	-	1	S/ 0.00	S/ 0.00
	Licencias o suscripciones de software	Entorno virtual de desarrollo de machine learning Colaboratory (Google)	cuenta	5	S/ 0.00	S/ 0.00
		SUBTOTAL				S/ 2,437.50
		IMPREVISTOS (10 %)				S/ 243.75
		TOTAL				S/ 2,681.25

Marco Teórico

Crianza de Cuyes

La ganadería es una actividad que consiste principalmente en la crianza, manejo y explotación de animales domesticables. Esta actividad se realiza con fines de producción de recursos que se pueden aprovechar de los animales. Dependiendo de la especie, se obtienen diversos recursos como cueros, carne, huevos, lana, miel, entre muchos otros productos derivados.

La ganadería en el Perú es primordialmente impulsada por el consumo y comercialización de carne y leche. En una buena parte de los Andes peruanos, también se lleva a cabo la ganadería tradicional de animales como vacas, ovejas, camélidos sudamericanos (llamas, alpacas, vicuñas) y cuyes. Sin embargo, un término más específico para referirse a la ganadería de cuyes es "crianza de cuyes".

La práctica de la crianza de cuyes se lleva a cabo debido a que la carne de cuy posee un valor biológico destacable, debido a que esta llega a contener los aminoácidos esenciales y ácidos grasos esenciales requeridos por el ser humano (Agencia-Peruana-de-Noticias, 2022). Es por esta razón que la crianza de cuyes está ampliamente distribuida a lo largo del país.

> En el Perú, país con la mayor población y consumo de cuyes, se registra una producción anual de 16 500 toneladas de carne proveniente del beneficio de más de 65 millones de cuyes, producidos por una población más o menos estable de 22 millones de animales criados básicamente con sistemas de producción familiar.
>
> (Food and Agriculture Organization, sf, Párr. 1)

Cuy *(Cavia Porcellus)*

"El cuy (cobayo o curí) (Véase la Figura 4) es un mamífero roedor originario de la zona andina de Bolivia, Colombia, Ecuador y Perú" (Chauca, 1997, Párr. 1). "Pertenece al género Cavia de la familia Caviidae. Además de que, por su capacidad de adaptación a

diversas condiciones climáticas, los cuyes pueden encontrarse desde la costa o el llano hasta alturas de 4 500 metros sobre el nivel del mar y en zonas tanto frías como cálidas" (Chauca, 1997, Párr. 2).

Debido a estas cualidades, los cuyes silvestres se han encontrado en grandes agrupaciones dispersas por América Central, el Caribe y Sudamérica hasta Paraguay, habitando en valles y pastizales. Sin embargo, su crianza doméstica solo abarca desde Venezuela hasta el noreste de Argentina y el norte de Chile. En general, estos animales son criados en jaulas o pozas, cuyo tamaño suele variar según la población de cuyes que la habita.

Figura 4

Ejemplares de cuyes de granja

Nota. Reproducido de Carne de cuy: estas son las bondades nutricionales de este alimento ancestral andino, por Agencia Peruana de Noticias [Fotografía], 2022, Andina (http://surl.li/gfjgy). Todos los derechos reservados 2023 por Agencia Peruana de Noticias. Reproducido con permiso del autor.

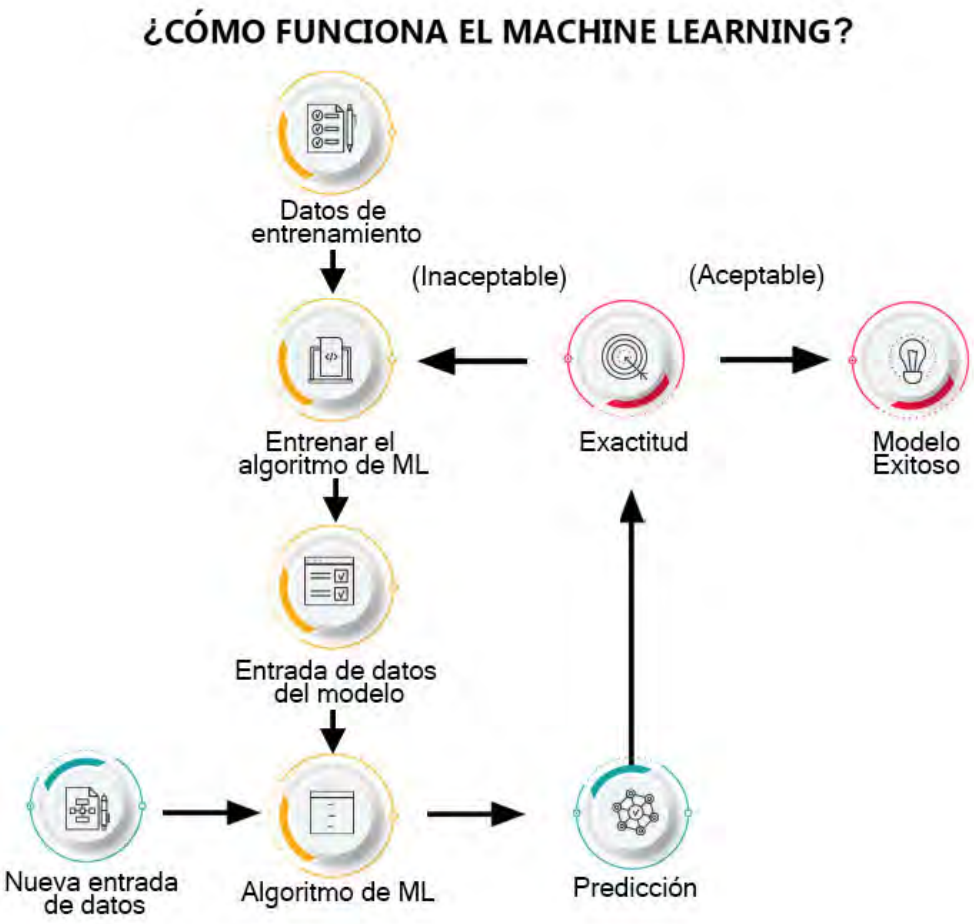
Machine Learning

El Aprendizaje Automático o ML es una disciplina de la Inteligencia Artificial (IA) que proporciona a las máquinas la capacidad de aprender automáticamente a partir de datos y experiencias pasadas, identificando patrones para hacer predicciones con una intervención humana mínima. Los métodos de aprendizaje automático permiten a los

ordenadores funcionar de forma autónoma sin programación explícita. Las aplicaciones de ML se alimentan con nuevos datos y pueden aprender, crecer, desarrollarse y adaptarse de forma independiente. El aprendizaje automático obtiene información reveladora a partir de grandes volúmenes de datos mediante algoritmos que identifican patrones y aprenden en un proceso iterativo. Los algoritmos de ML utilizan métodos de cálculo para aprender directamente de los datos en lugar de basarse en cualquier ecuación predeterminada que pueda servir como modelo (Kanade, 2022). A continuación, en la Figura 5 se puede observar a grandes rasgos cómo funciona el Machine Learning.

Figura 5

Caso de uso de alto nivel sobre cómo funciona el Machine Learning



Nota. Adaptado de What Is Machine Learning? Definition, Types, Applications, and Trends, por V. Kanade, 2022, Spiceworks (https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/). Todos los derechos reservados 2023 por Spiceworks Inc. Adaptado con permiso del autor.

Visión Artificial

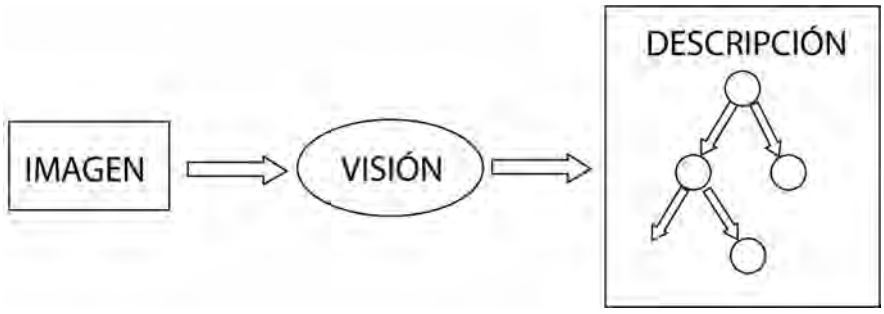
Desde el punto de vista formal, la visión artificial o visión por ordenador es un campo de la IA que permite a los ordenadores y sistemas obtener información significativa a partir de imágenes digitales, vídeos y otras entradas visuales, y emprender acciones o hacer recomendaciones basadas en esa información. Si la IA permite a los ordenadores pensar, la visión por ordenador les permite ver, observar y comprender (IBM, sfb).

Para entender la visión artificial, primero debemos comprender que la visión permite al ser humano percibir y comprender el mundo que le rodea. La visión artificial pretende duplicar este efecto, percibiendo y comprendiendo electrónicamente una imagen (Sonka *et al.*, 2013). En pocas palabras, la comprensión de una imagen por un sistema de visión artificial puede considerarse como el desentrañamiento de la información simbólica utilizando modelos construidos con ayuda de la geometría, la física, la estadística y la teoría del aprendizaje.

En conclusión, como se observa en la Figura 6, la visión artificial o visión por ordenador es un proceso análogo a la visión humana cuyo objetivo es extraer características del mundo real a través de las imágenes digitales, transformar estas características en información numérica o simbólica, y finalmente crear una descripción o interpretación del mundo real comprendida por el computador (Sucar y Gómez, 2011).

Figura 6

Esquema general de visión por computadora



Nota. Adaptado de Visión Computacional (p. 2), por L. Sucar y G. Gómez, 2011, Instituto Nacional de Astrofísica, Óptica y Electrónica

Etapas del Proceso de Visión Artificial

Según González *et al.* (2006), las principales etapas del proceso de visión artificial son las siguientes:

1. La primera etapa consiste en la adquisición de la imagen digital, para lo cual se necesitan sensores y el hardware para digitalizar la señal producida por el sensor.
2. La siguiente etapa es el preprocesamiento de la imagen, para que el objetivo tenga mejores resultados.
3. La tercera etapa es la segmentación, el objetivo de esta es dividir la imagen en las partes que la constituyen o los objetos que la forman.
4. La cuarta etapa es la parametrización, o también conocida como selección de rasgos. En esta etapa se dedica a extraer rasgos que produzcan alguna información cuantitativa de interés o rasgos para diferenciar un objeto de otro.
5. La quinta y última etapa es acerca del reconocimiento y la interpretación. El reconocimiento es el proceso en el que se asigna una etiqueta a un objeto basado en la información que proporcionan los descriptores. La interpretación de este lleva a asignar significado al conjunto de objetos reconocidos.

Deep Learning

El Aprendizaje Profundo o Aprendizaje Profundo (DL) es un subcampo del Aprendizaje Automático (ML) y representa un conjunto de arquitecturas de redes neuronales que resuelven problemas complejos y punteros. El aprendizaje profundo se refiere a arquitecturas de redes neuronales que incluyen muchas capas y tienen la capacidad de aprender (a través del entrenamiento) a asignar una entrada, como una imagen, a una o más salidas, como por ejemplo una clasificación.

Según Jones (2022), algunos de los tipos de DNN más populares que existen son las redes neuronales recurrentes, las redes neuronales convolucionales y las redes de memoria larga a corto plazo (Long Short-Term Memory (LSTM)).

Redes Neuronales Convolucionales

Las Redes Neuronales Convolucionales o CNN son un tipo especializado de red neuronal para procesar datos con una topología conocida, similar a una cuadrícula. Algunos ejemplos son los datos de series temporales, que pueden considerarse una cuadrícula 1D que toma muestras a intervalos regulares, y los datos de imágenes, que pueden considerarse una cuadrícula 2D de píxeles. El nombre "Red Neuronal Convolucional" indica que la red emplea una operación matemática llamada convolución. La convolución es un tipo especializado de operación lineal. Las redes convolucionales son simplemente redes neuronales que utilizan la convolución en lugar de la multiplicación matricial general en al menos una de sus capas (Goodfellow *et al.*, 2016).

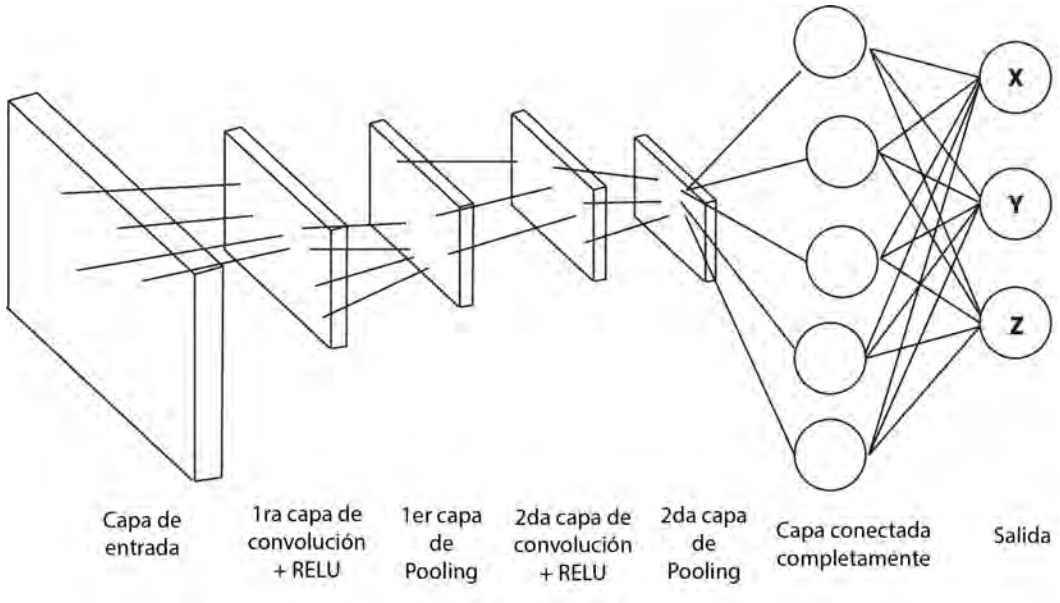
Arquitectura Básica de una CNN. El artículo de López-Ramos y Arco-García (2019) describe la estructura básica de una CNN, que consta de varias capas que realizan diversas operaciones sobre los datos de entrada, generalmente imágenes o secuencias. Estas capas incluyen:

- **Capa de entrada**: Es la capa que recibe el dato de entrada (imagen, audio, oración, etc.), que se representa por lo general como una matriz tridimensional de píxeles o valores según sea el caso. Cada píxel tiene un valor numérico que indica su intensidad o color.
- **Capa de convolución**: Esta capa es fundamental en una CNN y consiste en un conjunto de parámetros llamados filtros. Estos filtros tienen la misma forma que la entrada, pero son de menor dimensión. Durante el entrenamiento, cada filtro se desplaza sobre los datos de entrada y calcula un producto interno entre la entrada y el filtro. A este producto se le conoce como mapa de características debido a que el cálculo permite mapear las características del filtro y así extraer rasgos o patrones de la imagen, como bordes, formas, texturas, etc.
- **Capa de activación**: Es la capa que aplica una función no lineal sobre el mapa de características, introduciendo una no linealidad en la red. Esta capa permite que la

red aprenda funciones más complejas y mejore su capacidad de generalización. Una función de activación común es la función Rectified Linear Unit (ReLU) la cual consiste en neuronas con la función de activación de la forma $f(x) = max(0, x)$, aunque pueden ser seleccionadas otras funciones de activación como por ejemplo la tangente hiperbólica.

- **Capa de agrupación o pooling**: Esta capa reduce el tamaño del mapa de características resultante de la convolución. Se realiza mediante una operación de agregación, como el Max-pooling, que toma el valor máximo en una región de la imagen. Esto reduce la complejidad computacional, evita el sobreajuste y mejora la invariancia a las transformaciones de la imagen.
- **Capa de salida**: Esta capa produce la predicción final de la red, que puede ser una etiqueta de clase, un valor numérico o una probabilidad. Se compone generalmente de una capa densa o completamente conectada que recibe el vector de características resultante de las capas anteriores y aplica una función de activación adecuada para el problema, como la función Softmax en clasificación multiclase.

Figura 7
Arquitectura básica de una CNN



Nota. Adaptado de Aprendizaje profundo para la extracción de aspectos en opiniones textuales (p. 116), por D. López y L. Arco, 2019, Revista Cubana de Ciencias Informáticas.

Operación de Convolución. Según la definición proporcionada por Goodfellow *et al.* (2016), en su forma más general, la convolución es una operación que se realiza sobre dos funciones, ambas de un solo argumento con valores reales.

En las aplicaciones de aprendizaje automático, la entrada suele ser una matriz multidimensional de datos, y el kernel suele ser una matriz multidimensional de parámetros adaptados por el algoritmo de aprendizaje. Nos referiremos a estas matrices multidimensionales como tensores. Dado que cada elemento de la entrada y del kernel debe almacenarse explícitamente por separado, normalmente se asume que estas funciones son cero en todas partes excepto en el conjunto finito de puntos para los que se almacenan los valores. Esto significa que, en la práctica, se puede implementar la suma infinita como una suma sobre un número finito de elementos de matriz.

Además, a menudo se utilizan las convoluciones sobre más de un eje a la vez. Por ejemplo, si se usa una imagen bidimensional I como entrada, probablemente también se deba usar un kernel bidimensional K, pero como la convolución es conmutativa, se puede escribir de forma equivalente:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n) \tag{1}$$

Por lo general, esta última fórmula es más fácil de implementar en una biblioteca de ML, ya que hay menos variación en el rango de valores válidos de m y n.

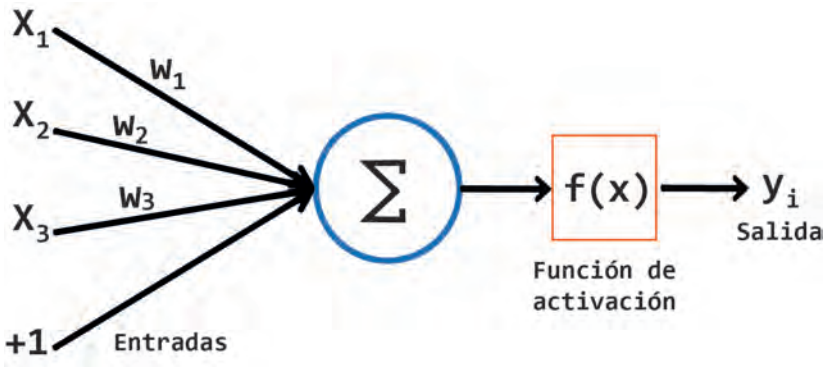
La propiedad conmutativa de la convolución surge porque se ha volteado el kernel en relación con la entrada, en el sentido de que a medida que m aumenta, el índice en la entrada aumenta, pero el índice en el kernel disminuye. La única razón para cortar el kernel es obtener la propiedad conmutativa. Aunque la propiedad conmutativa es útil para escribir pruebas, no suele ser una propiedad importante de la implementación de una red neuronal. En su lugar, muchas bibliotecas de redes neuronales implementan una función relacionada llamada correlación cruzada, que es lo mismo que la convolución, pero sin voltear el kernel.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n) \tag{2}$$

Funciones de Activación. De una manera similar al cerebro humano, la unidad básica en una red neuronal se denomina neurona por lo que su función se asemeja a la de una neurona humana, ya que recibe entradas específicas y genera una salida correspondiente. Desde un punto de vista puramente matemático, en el contexto del aprendizaje automático, una neurona se emplea como un componente que representa una función matemática y su único propósito es producir una salida aplicando esta función a las entradas dadas. En términos más concisos, la función matemática utilizada en una neurona se denomina comúnmente función de activación (Véase la Figura 8).

Figura 8

Esquema de una neurona artificial



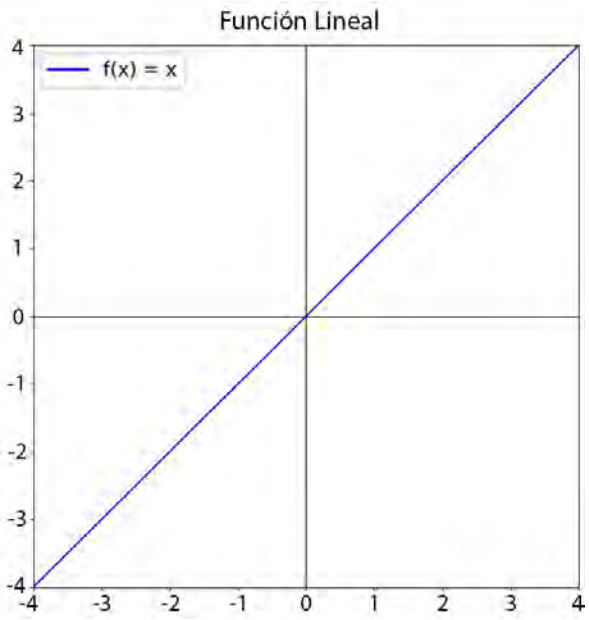
Si no se emplea una función de activación en una red neuronal, la salida sería simplemente una función lineal básica, limitada a relaciones lineales. Aunque las ecuaciones lineales son simples, no pueden manejar mapeos complejos en los datos. Una red neuronal sin funciones de activación se asemeja a un modelo de regresión lineal, con un rendimiento limitado. Sin embargo, para tareas más complejas como el procesamiento de imágenes, voz y otros datos de alta dimensión y no lineales, se utilizan funciones de activación y técnicas de redes neuronales profundas (Deep Learning). Estas redes tienen capas ocultas y una arquitectura compleja que les permite aprender y extraer conocimiento de datos complicados, lo cual es su objetivo principal (Sharma *et al.*, 2020).

Algunas de las funciones de activación más comunes son las siguientes:

Función Lineal. La función de activación lineal enunciada en la Ecuación 3 es directamente proporcional a la entrada (Ver Figura 9).

$$f(x) = x \tag{3}$$

Figura 9
Función de activación lineal



El uso de una función lineal no proporciona muchas ventajas, ya que la red neuronal no sería capaz de mejorar el error de manera efectiva. Esto se debe a que, al derivar una función lineal, se obtiene un valor constante para el gradiente en cada iteración, lo que dificulta que la red identifique patrones complejos en los datos. En consecuencia, las funciones lineales son más adecuadas cuando se busca interpretabilidad y se aplican en tareas simples.

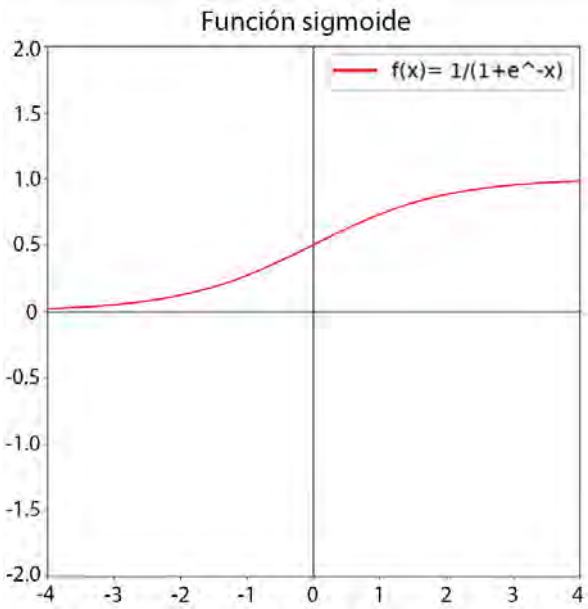
Función Sigmoide. Es la función de activación más utilizada ya que es una función no lineal. La función sigmoidea transforma los valores en el rango de 0 a 1. Se puede definir como:

$$\frac{1}{1+e^{-x}} \tag{4}$$

La aplicación de la función sigmoide es ideal en modelos donde se necesita predecir probabilidades como salida, siempre que estas probabilidades estén en un rango de 0 a 1. La función sigmoide se representa mediante una curva en forma de S. (Ver Figura 10).

Figura 10

Función de activación sigmoide

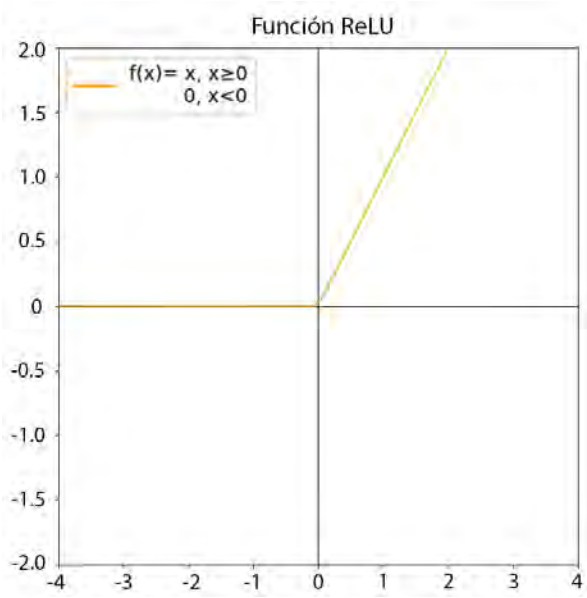


Función ReLU. La función de activación ReLU es una función no lineal ampliamente empleada en las redes neuronales. Una de las ventajas notables de usar la función ReLU es que no todas las neuronas se activan simultáneamente. Esto implica que una neurona se desactiva solo cuando la salida de la transformación lineal es igual a cero. Su definición matemática es la siguiente:

$$f(x) = max(0, x) \tag{5}$$

ReLU(Ver Figura 11) es más eficiente que otras funciones porque no todas las neuronas se activan al mismo tiempo, sino que se activa un cierto número de neuronas a la vez.

Figura 11
Función de activación ReLU



Funciones de Agrupación o Pooling. El resultado de las capas de convolución y activación se denomina mapa de características. Este mapa se utiliza como entrada en la siguiente capa, que recibe el nombre de capa de agrupación o pooling debido a que en esta se aplica una función conocida como pooling. Según Goodfellow *et al.* (2016), la función de agrupación tiene como objetivo reemplazar la salida de la red en un punto específico con un resumen estadístico de las salidas cercanas. Esto se logra seleccionando valores representativos de un conjunto de valores adyacentes en el mapa de características, lo que conduce a una reducción en el tamaño y la complejidad de la representación sin perder información crucial.

Algunas de las funciones locales y globales de agrupamiento más usadas son:

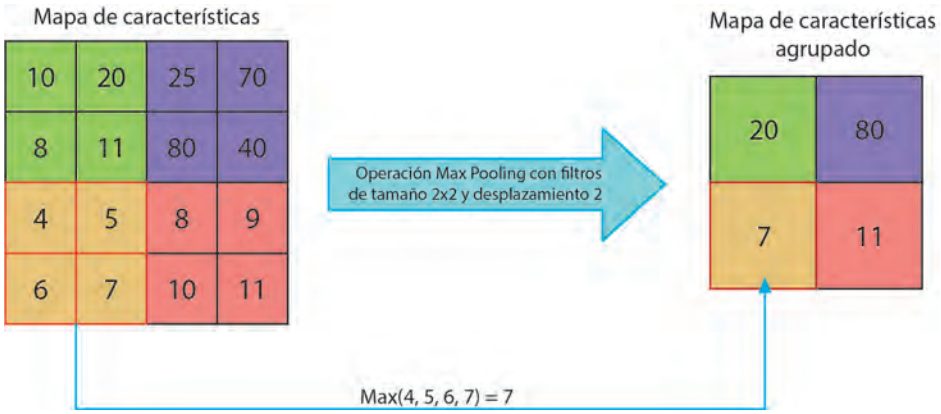
Max Pooling. El método max pooling (Boureau *et al.*, 2010) es sencillo y se aplica ampliamente en CNN porque no hay parámetros que necesiten ser ajustados. La operación de max pooling se muestra en la Figura 12 para una mejor comprensión.

En palabras de Zafar *et al.* (2022), max pooling es un mecanismo que optimiza el tamaño espacial de un mapa de características a la vez que proporciona a la red invariancia de traslación. Esto se realiza mostrando el mayor valor en el mapa de características

principalmente dentro de un vecindario de tamaño $k \times k$. La técnica de max pooling identifica el elemento más grande en cada región de pooling (Singh *et al.*, 2021).

Figura 12

Ejemplo de la operación Max Pooling



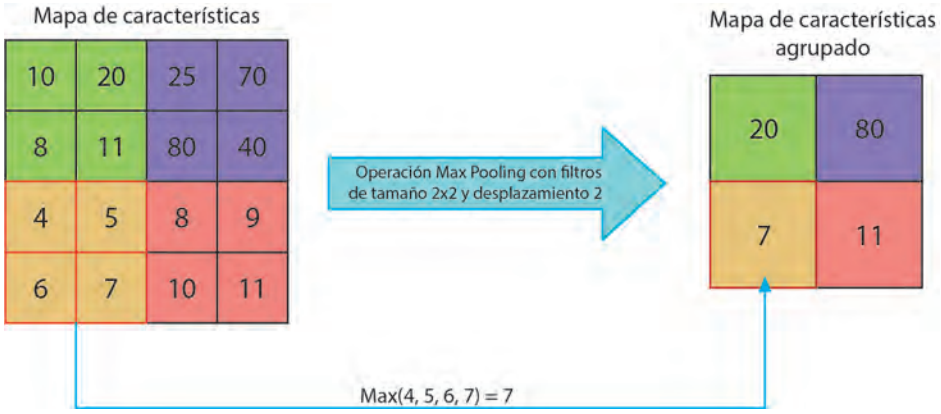
Nota. Adaptado de A Comparison of Pooling Methods for Convolutional Neural Networks (p. 4), por A. Zafar et al., 2022, Applied Sciences, 12(17):8643. CC BY 4.0

Average Pooling. El procedimiento de Average Pooling, tal como lo describe Zafar *et al.* (2022), opera de la siguiente manera: la imagen de entrada se fragmenta en múltiples áreas rectangulares individuales. Luego, se determina el promedio de los valores dentro de cada una de estas áreas rectangulares y se genera el canal de salida correspondiente. La operación de Average Pooling se ilustra visualmente en la Figura 13 para una mejor comprensión.

El inconveniente de este método es que conduce a una disminución de la información en términos de contraste. Al estimar la media, se tienen en cuenta todos los valores de activación de la caja rectangular. En efecto, la media estimada será baja si la intensidad de todas las funciones de activación es baja, lo que provocará una disminución del contraste. El problema empeorará cuando la mayoría de las activaciones de la zona de agrupación tengan un valor cero. En esa situación, la característica convolucional se reduciría significativamente (Zhang *et al.*, 2018).

Figura 13

Ejemplo de la operación Average Pooling



Nota. Adaptado de A Comparison of Pooling Methods for Convolutional Neural Networks (p. 5), por A. Zafar et al., 2022, Applied Sciences, 12(17):8643. CC BY

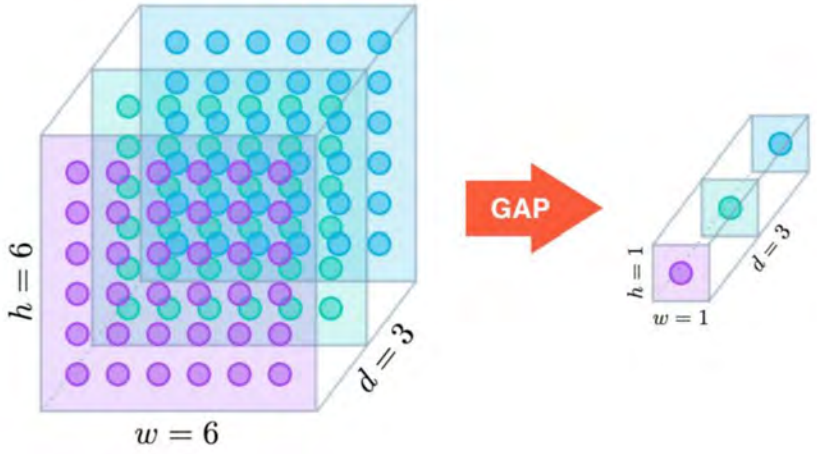
Global Average Pooling. La operación Global Average Pooling (GAP) se utiliza comúnmente para disminuir la complejidad de los mapas de características generados por las capas convolucionales anteriores en una red neuronal. En palabras de Hsiao *et al.* (2019), la operación de GAP implica tomar un mapa de características y calcula el valor promedio de cada canal (o característica) en el mapa de características. En otras palabras, toma la media de todos los valores en un canal para generar un solo valor. Esto se hace para cada canal en el mapa de características.

Una de las ventajas principales de la capa de GAP es su capacidad para significativamente reducir la cantidad de parámetros en la red en comparación con las capas completamente conectadas convencionales. Además, al calcular este promedio, contribuye a que la red sea más resistente al sobreajuste, minimizando la posibilidad de que la red memorice los datos de entrenamiento (Senthil Pandi *et al.*, 2022).

Las capas de GAP llevan a cabo una reducción dimensionalidad a un nivel extremo gracias a que de un tensor con dimensiones $h \times w \times d$ se reduce a dimensiones de $1 \times 1 \times d$. Esto se logra al reducir cada mapa de características $h \times w$ a un solo número mediante la obtención del promedio de todos los valores hw (Shustanov y Yakimov, 2019). Para tener una mejor comprensión de cómo funciona esta operación, se puede consultar la Figura 14 en la que se presenta un ejemplo visual de su funcionamiento.

Figura 14

Ejemplo de la operación Global Average Pooling



Nota. Reproducido de Modification of single-purpose CNN for creating multi-purpose CNN (p. 4), por A. Shustanov y P. Yakimov, 2019, Journal of Physics Conference, 1368(5):052036. CC BY 3.0

Capa Densa o Totalmente Conectada. Una capa densa es una capa completamente conectada en la que cada neurona de la capa anterior está conectada a cada neurona de la capa densa. En una CNN típica, las capas densas suelen estar ubicadas al final de la red después de varias capas de convolución y agrupación. Estas capas densas se utilizan para realizar la clasificación final de los datos procesados por la red. Cada neurona en una capa densa realiza una combinación lineal de las salidas de las capas anteriores y aplica una función de activación no lineal, como la función sigmoide para producir la salida final (IBM, sfa).

La capa densa es esencial para mapear las características aprendidas por las capas convolucionales en las clases o salidas de interés en una determinada tarea. Es en esta capa donde se toman las decisiones finales sobre las predicciones o categorización de los datos de entrada.

Aplicaciones de las CNN. Las CNN se utilizan en una amplia gama de aplicaciones en diversos campos y han demostrado ser altamente efectivas en una variedad de tareas, que incluyen:

Clasificación de Imágenes. La clasificación de imágenes es una de las aplicaciones

esenciales de las CNN, donde estas redes tienen la capacidad de aprender y reconocer características visuales complejas para etiquetar imágenes.

Segmentación Semántica. En la segmentación semántica, las CNN van un paso más allá de la clasificación de imágenes, ya que asignan una etiqueta a cada píxel en una imagen. Esto posibilita la comprensión de cómo están distribuidos los objetos y las regiones de interés en una escena determinada.

Detección de Objetos. La detección de objetos conlleva la tarea de encontrar y categorizar varios objetos en una imagen. Las CNN pueden reconocer la presencia de objetos y trazar rectángulos delimitadores alrededor de ellos.

Regresión. A pesar de que las CNN están principalmente diseñadas para tareas de clasificación y procesamiento de imágenes, también son aplicables a problemas de regresión, donde se busca predecir un valor numérico en lugar de una etiqueta.

Métricas de Evaluación del Rendimiento de CNN de Regresión

En las tareas de regresión, el objetivo principal consiste en minimizar la disparidad entre la estimación y el valor verdadero. Las variables utilizadas en estas métricas son: T_i, que representa el valor verdadero asociado con la i-ésima imagen, y Y_i, que representa el valor estimado para la i-ésima imagen.

Error Cuadrático Medio (MSE). El MSE es utilizada principalmente como función de pérdida (loss) durante el entrenamiento de los modelos. El Error Cuadrático Medio se calcula como el promedio del cuadrado del error residual en el conjunto de entrenamiento. Para N imágenes de entrenamiento, la función de pérdida MSE se define como:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - T_i)^2 \tag{6}$$

Error Absoluto Medio (MAE). El MAE es una de las métricas utilizadas para evaluar y comparar el rendimiento de las CNN. Esta métrica mide la magnitud promedio de

los errores de predicción en las unidades originales. En otras palabras, proporciona el promedio de la desviación de las predicciones respecto a los valores reales, sin considerar la escala o proporción. Se expresa como la norma de la medida del error entre dos variables aleatorias continuas. Para N predicciones Y_i y sus correspondientes objetivos T_i, MAE viene dado por:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|Y_i - T_i| \tag{7}$$

Error Absoluto Porcentual Medio (MAPE). El MAPE es la segunda métrica utilizada para la evaluación y comparativa de los modelos. Esta métrica mide el error de predicción promedio en términos de porcentaje en lugar de unidades originales, lo que hace que el MAPE sea más interpretable, ya que calcula cuánto se desvían las predicciones en promedio como un porcentaje del valor real. Se expresa como la porcentualización de la norma del cociente del error absoluto medio entre el valor de la verdad fundamental (T_i). Para N predicciones, se define como:

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{Y_i - T_i}{T_i}\right| \tag{8}$$

Arquitectura ResNet

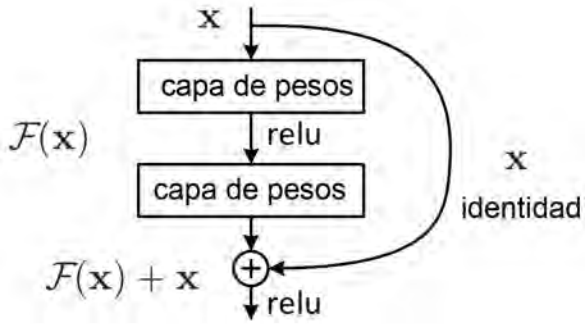
En el trabajo "Deep Residual Learning for Image Recognition" (He *et al.*, 2015), se aborda la problemática del entrenamiento de redes neuronales profundas mediante la introducción de conexiones residuales. La motivación principal surge de la observación de que el rendimiento de las redes tradicionales disminuye a medida que aumenta la profundidad, debido al desafío del desvanecimiento del gradiente.

La arquitectura ResNet propone bloques residuales (Ver Figura 15). Cada bloque viene compuesto por dos capas de convolución seguidas por una conexión residual. La principal contribución radica en la conexión residual misma, que actúa como un "atajo" que permite que la información original fluya sin obstáculos a través del bloque. Esta conexión

se modela como la suma de la entrada original a la salida del bloque, permitiendo que los gradientes se propaguen directamente a través de la red.

Figura 15

Esquema de un bloque residual básico



Nota. Adaptado de Deep Residual Learning for Image Recognition (p. 2), por K. He et al., 2015, arXiv, 1512.03385v1. CC BY

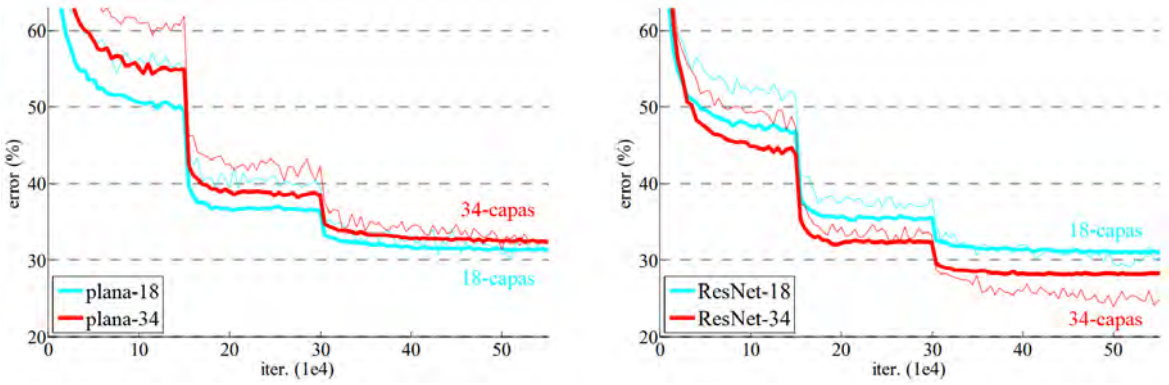
La variante más simple y eficaz de la conexión residual es la "identity shortcut connection" donde la entrada se suma directamente a la salida. Esta estrategia fue diseñada con la intención de permitir que la red aprenda transformaciones residuales, es decir, las diferencias entre las activaciones previas y posteriores al bloque. Además, estas conexiones no introducen parámetros adicionales ni complejidad computacional, lo que resulta en redes menos complejas que las arquitecturas planas.

Otro aspecto destacado del estudio es la demostración experimental (Véase la Figura 16) en donde las ResNets más profundas logran superar a sus contrapartes menos profundas y a otras arquitecturas planas basadas en la filosofía de VGG en términos de Accuracy en la clasificación de imágenes en los dataset ImageNet y CIFAR-10. Además, se presenta un análisis detallado del comportamiento del entrenamiento y el efecto de la profundidad en la convergencia y la generalización del modelo, logrando así resolver en gran medida el problema de la degradación de las redes más profundas.

Modelo ResNet50. De las variantes proporcionadas de la arquitectura ResNet se escoge la variante con 50 capas por ser de una complejidad apropiada para datasets pequeños y haber demostrado un rendimiento más que apropiado en comparación a las variantes más profundas de este tipo de arquitectura.

Figura 16

Resultados de las pruebas de entrenamiento en ImageNet



Nota. Izquierda: Redes planas de 18 y 34 capas. Derecha: ResNets de 18 y 34 capas. Las curvas delgadas denotan el error de entrenamiento y las curvas en negrita denotan el error de validación. Adaptado de Deep Residual Learning for Image Recognition (p. 5), por K. He et al., 2015, arXiv, 1512.03385v1. CC BY

Para la implementación de esta variante se reemplazan los bloques de dos capas del modelo de 34 capas con un bloque de cuello de botella con 3 capas (véase el Cuadro 2)

Cuadro 2

Arquitecturas ResNet para ImageNet

Nombre de la capa	Tamaño de salida	34-capas	50-capas
conv1	112×112	7×7, 64, stride 2	
conv2_x	56×56	3×3, max pool, stride 2	
		$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax	
FLOPS		$3{,}6 \times 10^9$	$3{,}8 \times 10^9$

Nota. Los bloques de construcción se muestran entre corchetes. El downsampling se realiza mediante conv3_1, conv4_1 y conv5_1 con un stride de 2. Adaptado de Deep Residual Learning for Image Recognition (p. 5), por K. He et al., 2015, arXiv, 1512.03385v1. CC BY

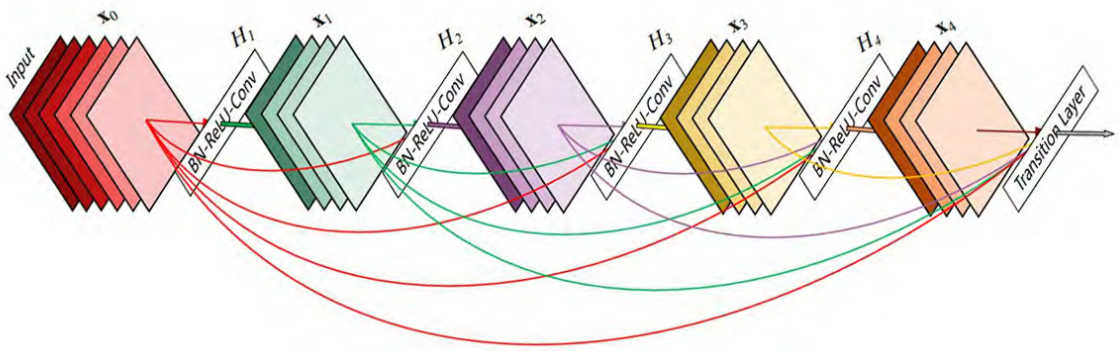
Arquitectura DenseNet

DenseNet, una innovadora arquitectura, presentada en el paper “Densely Connected Convolutional Networks” (Huang *et al.*, 2018), aborda desafíos fundamentales en el diseño de redes neuronales al proponer conexiones densas en lugar de las conexiones convencionales de capa a capa.

La idea central que subyace en DenseNet es el concepto de conexiones densas entre capas. A diferencia de las estructuras convencionales que establecen conexiones de capa a capa, DenseNet establece una conexión directa entre cada capa y todas las capas subsiguientes. En otras palabras, la salida de una capa se convierte en la entrada de todas las capas subsiguientes dentro de un mismo bloque (Ver Figura 17). Este diseño intra-bloque refuerza la conexión densa, promoviendo así una propagación de información más eficiente. En consecuencia, esta conectividad densa presenta beneficios significativos en términos de facilitar el entrenamiento, mejorar la eficiencia de los parámetros y potenciar la capacidad de aprendizaje de características complejas.

Figura 17

Un bloque denso de 5 capas y tasa de crecimiento $k = 4$



Nota. Adaptado de Convolutional Networks with Dense Connectivity (p. 1), por G. Huang et al., 2018, arXiv, 2001.02394v1. CC BY 4.0

Otra de las virtudes de DenseNet es que se centra en mejorar el flujo de gradientes. Las conexiones directas entre capas abordan el desafío del desvanecimiento de gradientes al permitir un flujo de información y gradientes más eficiente durante el proceso de entrenamiento. Esto se logra al proporcionar a cada capa acceso directo a los gradientes de

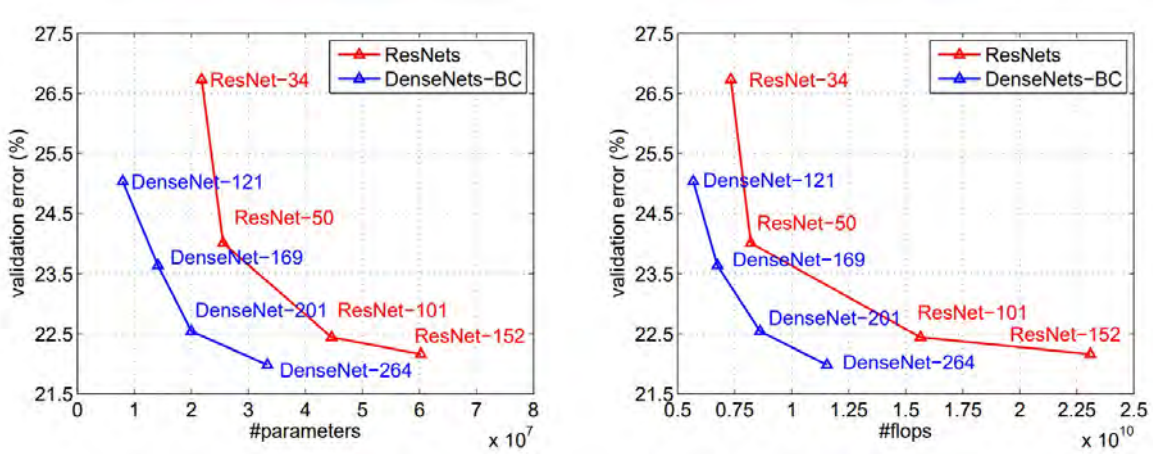
la función de pérdida y a la señal de entrada original, generando así una supervisión profunda implícita. Este enfoque contribuye de manera significativa al entrenamiento de arquitecturas de red más profundas. Además, se ha observado que las conexiones densas también ejercen un efecto regularizador, reduciendo el riesgo de sobreajuste, especialmente en escenarios donde se trabaja con conjuntos de entrenamiento pequeños.

Adicionalmente, la arquitectura promueve la reutilización de características de manera efectiva. Al tener conexiones directas entre capas, cada capa tiene acceso a las salidas de todas las capas anteriores, lo que fomenta la creación de representaciones más ricas y complejas con menos parámetros. En la palabras de Huang *et al.* (2018), la concatenación de mapas de características aprendidos por distintas capas aumenta la variación en la entrada de las capas siguientes y mejora la eficiencia en comparación a los modelos ResNet (Ver Figura 18).

Esta eficiencia en la reutilización de características contribuye a la capacidad única de DenseNet para lograr un rendimiento notable con una cantidad significativamente menor de parámetros en comparación con otras arquitecturas como ResNet.

Figura 18

Comparación de las tasas de error Top-1 de DenseNets y ResNets en el dataset de validación ImageNet



Nota. Comparación en función de los parámetros aprendidos (izquierda) y de los FLOPs durante el tiempo de prueba (derecha). Adaptado de Convolutional Networks with Dense Connectivity (p. 7), por G. Huang et al., 2018, arXiv, 2001.02394v1. CC BY 4.0

Modelo DenseNet121. De los modelos ofrecidos dentro de la arquitectura DenseNet, se elige el modelo DenseNet121 que es un modelo adecuado para tareas con un conjunto de entrenamiento pequeño, además de ser una arquitectura con muy poca complejidad computacional ideal para la regresión.

La estructura del modelo se describe en el Cuadro 3 a continuación:

Cuadro 3

Arquitecturas DenseNet para ImageNet

Capas	Tamaño de salida	DenseNet-121
Convolución	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Bloque denso (1)	56×56	$\begin{bmatrix} 1 \times 1, conv \\ 3 \times 3, conv \end{bmatrix} \times 6$
Capa de Transición (1)	56×56 28×28	1×1 conv 2×2 average pool, stride 2
Bloque denso (2)	28×28	$\begin{bmatrix} 1 \times 1, conv \\ 3 \times 3, conv \end{bmatrix} \times 12$
Capa de Transición (2)	28×28 14×14	1×1 conv 2×2 average pool, stride 2
Bloque denso (3)	14×14	$\begin{bmatrix} 1 \times 1, conv \\ 3 \times 3, conv \end{bmatrix} \times 24$
Capa de Transición (3)	14×14 7×7	1×1 conv 2×2 average pool, stride 2
Bloque denso (4)	7×7	$\begin{bmatrix} 1 \times 1, conv \\ 3 \times 3, conv \end{bmatrix} \times 16$
Capa de clasificación	1×1	7×7 global average pool 1000D totalmente conectadas, softmax

Nota. La tasa de crecimiento para todas las redes es $k = 32$. Nótese que cada capa "conv" mostrada en la tabla corresponde a la secuencia BN-ReLU-Conv. Adaptado de Convolutional Networks with Dense Connectivity (p. 5), por G. Huang et al., 2018, arXiv, 2001.02394v1. CC BY 4.0

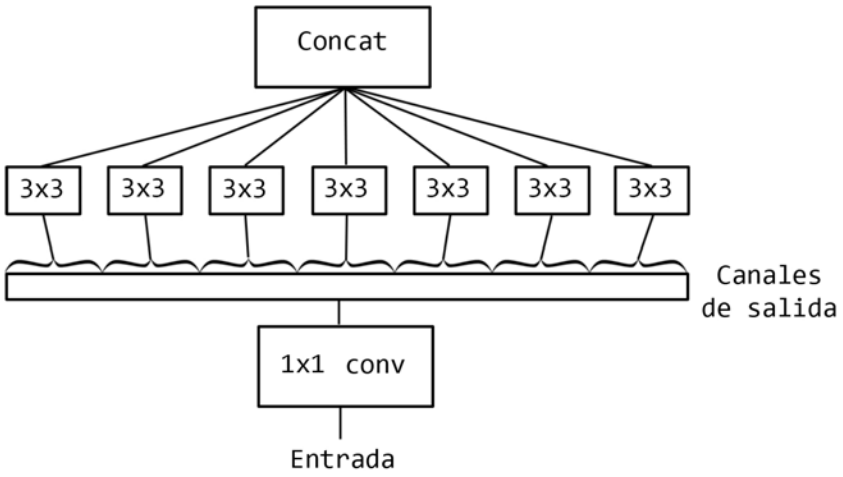
Arquitectura Xception

En el contexto del desafío inherente a la eficiencia computacional en las Redes Neuronales Convolucionales (CNN), el artículo "Xception: Deep Learning with Depthwise Separable Convolutions" Chollet (2017), presenta una solución innovadora siguiendo la siguiente hipótesis: "el mapeo de las correlaciones entre canales y las correlaciones espaciales en los mapas de características de las redes neuronales convolucionales puede desacoplarse por completo".

Este modelo se inspira en una "versión extrema" creada a partir de un módulo Inception (Szegedy *et al.*, 2014) (Véase la Figura 19). "Xtreme Inception" o solamente "Xception" aborda la eficiencia de manera diferente, optando por reemplazar los módulos Inception de la arquitectura con el mismo nombre por convoluciones separables en profundidad.

Figura 19

Esquema de una versión extrema a partir de un módulo Inception



Nota. Adaptado de Xception: Deep Learning with Depthwise Separable Convolutions (p. 2), por F. Chollet, 2017, arXiv, 1610.02357v3. CC BY 4.0

Siguiendo la hipótesis planteada con anterioridad, se puede decir que la contribución central de Xception radica en su arquitectura basada en "convoluciones separables en profundidad". Es decir que, en lugar de utilizar convoluciones estándar, el modelo propone dividir las convoluciones en dos etapas: una convolución en profundidad aplicada independientemente sobre cada canal de la entrada y una convolución puntual (convolución de 1×1), que proyecta los canales de salida de la convolución en profundidad en un nuevo espacio de canales. Este enfoque reduce significativamente el número de parámetros y operaciones en comparación con las arquitecturas convencionales.

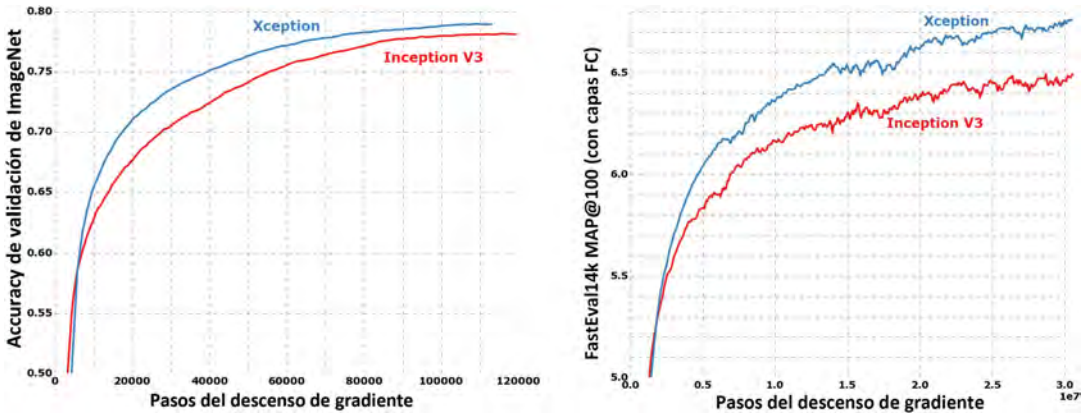
La técnica de convoluciones separables en profundidad que implica realizar convoluciones espaciales y en profundidad por separado, da como resultado una disminución drástica en el costo computacional. Más específicamente, la arquitectura de Xception organiza estas convoluciones en bloques, donde cada bloque utiliza

convoluciones separables en profundidad (Ver Figura 21). Estos bloques se apilan para formar la estructura completa de la red. Además, para reducir aún más la dimensionalidad espacial incorpora capas de pooling y conexiones residuales propuestas en el trabajo de He *et al.* (2015) con el objetivo de facilitar el flujo de gradiente entre capas.

En términos de resultados, los experimentos realizados en base a los datasets ImageNet y JFT, demuestran que Xception logra un rendimiento competitivo en comparación con Inception V3 que fue elegida para la comparación por tener un número similar de parámetros (Ver Figura 20). Esta eficiencia hace que Xception sea atractivo para aplicaciones en entornos con recursos limitados.

Figura 20

Perfil de entrenamiento de Xception



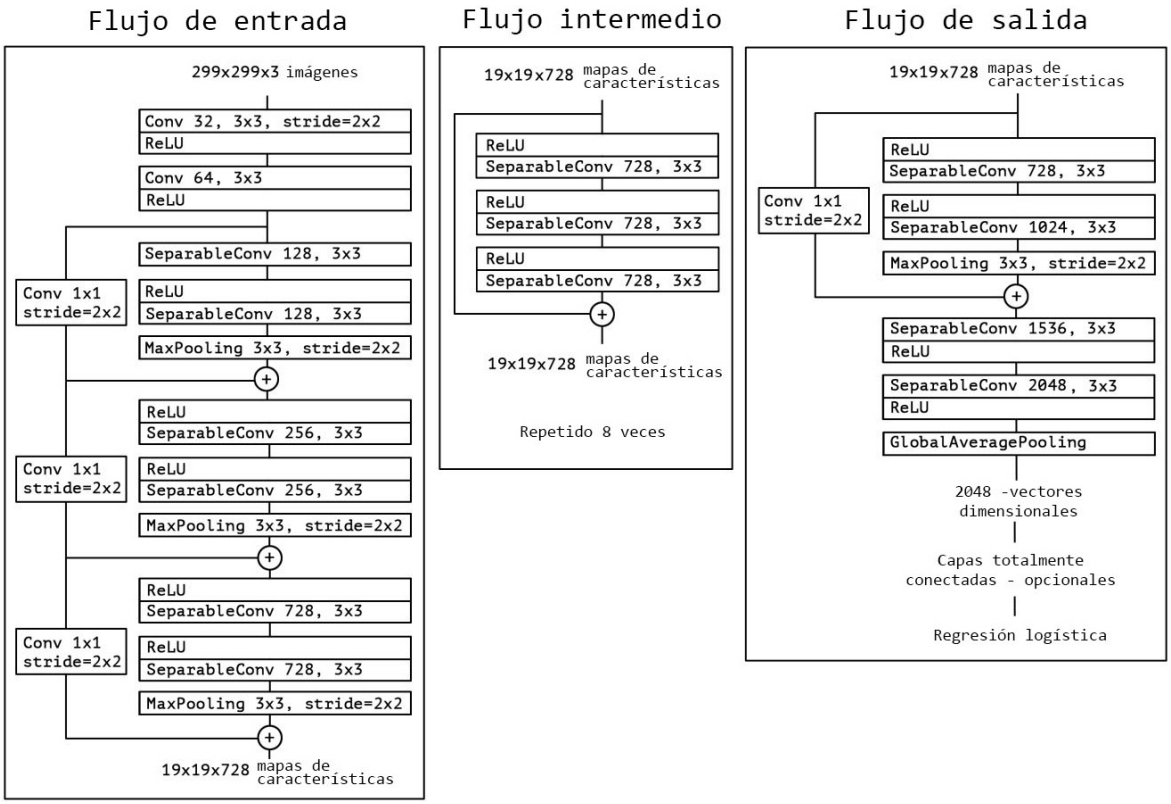
Nota. Izquierda: Curvas de validación en ImageNet. Derecha: Curvas de validación en JFT con capas totalmente conectadas. Adaptado de Xception: Deep Learning with Depthwise Separable Convolutions (p. 6), por F. Chollet, 2017, arXiv, 1610.02357v3. CC BY 4.0

Descripción de la Arquitectura Xception. La arquitectura Xception (Ver Figura 21) tiene 36 capas convolucionales que forman la base de extracción de características de la red. Estas están estructuradas en 14 módulos, todos ellos con conexiones residuales lineales a su alrededor, excepto el primer y el último módulo.

En resumen, la arquitectura Xception es una pila lineal de capas convolucionales separables en profundidad con conexiones residuales (Chollet, 2017).

Figura 21

La arquitectura Xception



Nota. Los datos pasan primero por el flujo de entrada, luego por el flujo intermedio, que se repite ocho veces, y finalmente por el flujo de salida. Adaptado de Xception: Deep Learning with Depthwise Separable Convolutions (p. 5), por F. Chollet, 2017, arXiv, 1610.02357v3. CC BY 4.0

Por último, a continuación se ofrece un cuadro de estadísticas que destaca aspectos clave de las variantes seleccionadas de cada arquitectura revisada.

En el Cuadro 4, las métricas Top-1 Accuracy y Top-5 Accuracy evalúan el desempeño del modelo en el conjunto de datos de validación ImageNet para tareas de clasificación. En cuanto a la profundidad, se refiere a la estructura topológica completa de la red, incluyendo capas de activación, capas de normalización de lotes, entre otras. La medida de profundidad tiene en cuenta el número total de capas con parámetros. Respecto al tiempo por paso de inferencia, se calcula como el promedio de 30 lotes realizados en 10 repeticiones (Chollet, 2015).

Cuadro 4

Estadísticas de las redes ResNet50, DenseNet121 y Xception

Modelo	Tamaño (MB)	Top-1 Accuracy	Top-5 Accuracy	Parámetros	Profundidad	Tiempo por paso de inferencia (CPU)	Tiempo por paso de inferencia (GPU)
ResNet50	98	74.90 %	92.10 %	25.6M	107	58.2 ms	4.6 ms
DenseNet121	33	75.00 %	92.30 %	8.1M	242	77.1 ms	5.4 ms
Xception	88	79.00 %	94.50 %	22.9M	81	109.4 ms	8.1 ms

Nota. Adaptado de Keras Applications, por F. Chollet, 2015, Keras (https://keras.io/api/applications/). Todos los derechos reservados 2023 por F. Chollet. Adaptado con permiso del autor.

Aumento de Datos

El aumento de datos es una técnica para aumentar artificialmente el conjunto de entrenamiento mediante la creación de copias modificadas de un conjunto de datos utilizando datos existentes. Incluye realizar pequeños cambios en el conjunto de datos o utilizar el aprendizaje profundo para generar nuevos puntos de datos.

El aumento de datos es un método muy potente para conseguirlo. Los datos aumentados representarán un conjunto más completo de posibles puntos de datos, minimizando así la distancia entre el conjunto de entrenamiento y el de validación, así como cualquier conjunto de pruebas futuro (Shorten y Khoshgoftaar, 2019).

Entre las técnicas tradicionales de aumento de datos se incluyen la rotación aleatoria, reescalado, volteado vertical y horizontal, traslación, recortado, ampliación, oscurecimiento y aumento del brillo o modificación del color, escalado en grises, cambio del contraste, adición de ruido, borrado de zonas aleatorias y cambio de la perspectiva de proyección.

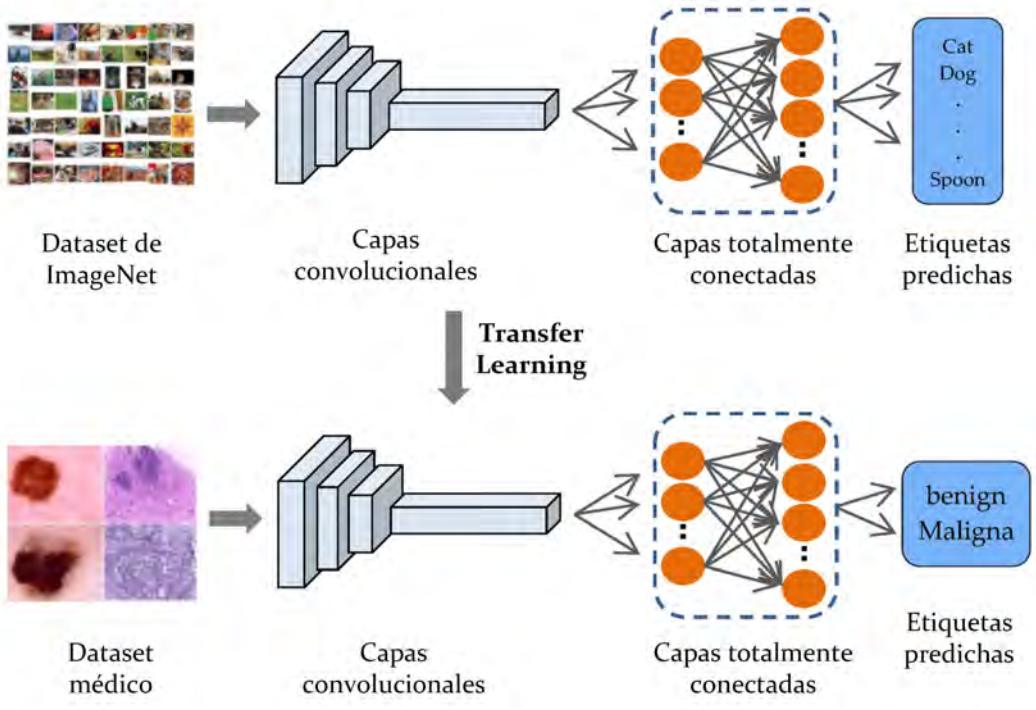
Transfer Learning

El transfer learning o aprendizaje por transferencia es un enfoque en el campo del aprendizaje automático y la inteligencia artificial. Se basa en el hecho de que las personas puedan aplicar de forma inteligente los conocimientos aprendidos previamente para resolver nuevos problemas más rápidamente o con mejores soluciones (Pan y Yang, 2010).

De manera análoga, los modelos de aprendizaje automático que pueden aprender representaciones generales y útiles de datos en una tarea y luego transferir ese conocimiento a otra tarea relacionada. A continuación en la Figura 22 se puede observar un esquema del aprendizaje por transferencia:

Figura 22

Transfer Learning de ImageNet



Nota. Adaptado de Incorporating a Novel Dual Transfer Learning Approach for Medical Images (p. 2), por A. Mukhlif et al., 2023, Sensors, 23(2):570. CC BY 4.0

Implementaciones del Transfer Learning

Según Kc *et al.* (2021), el transfer learning se lleva a cabo generalmente de dos maneras:

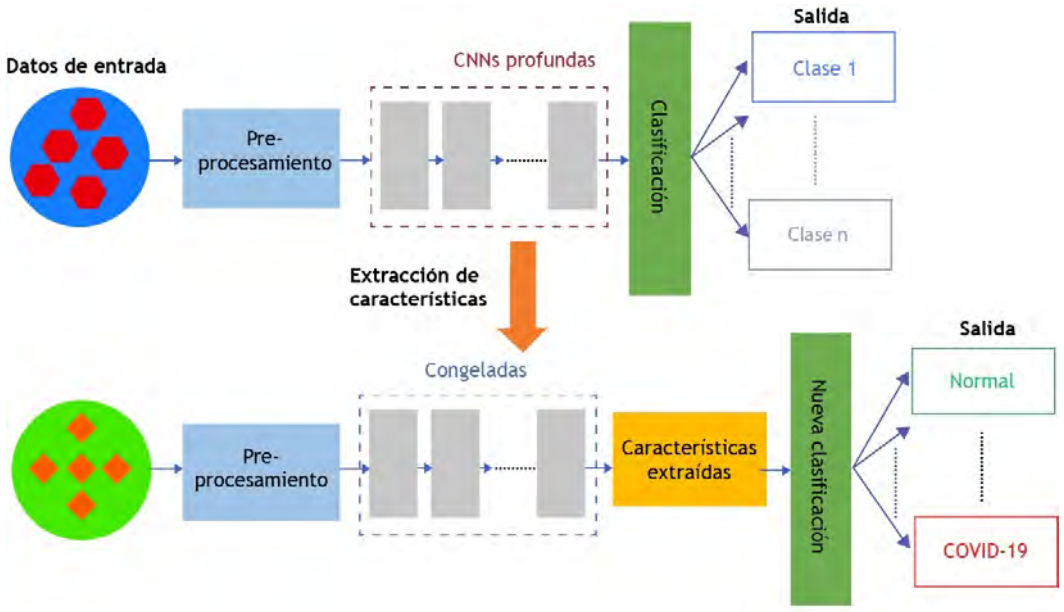
Extracción de Características (Feature extraction). En este enfoque, el modelo preentrenado se emplea para extraer características relevantes de los datos de entrada. Luego, estas características se utilizan como entrada para un nuevo modelo diseñado específicamente para la tarea en cuestión (Véase la Figura 23).

El extractor de características se implementa al reemplazar la capa de salida final por un clasificador apropiado y al congelar todos los pesos de la red, excepto los de la capa

final totalmente conectada, cuyas neuronas están conectadas completamente a todas las activaciones de la capa precedente. El resto de la red se mantiene inmutable como un extractor de características constante, y las características reutilizables se aprovechan completamente del preentrenamiento (Kc *et al.*, 2021).

Figura 23

Esquema de la extracción de características



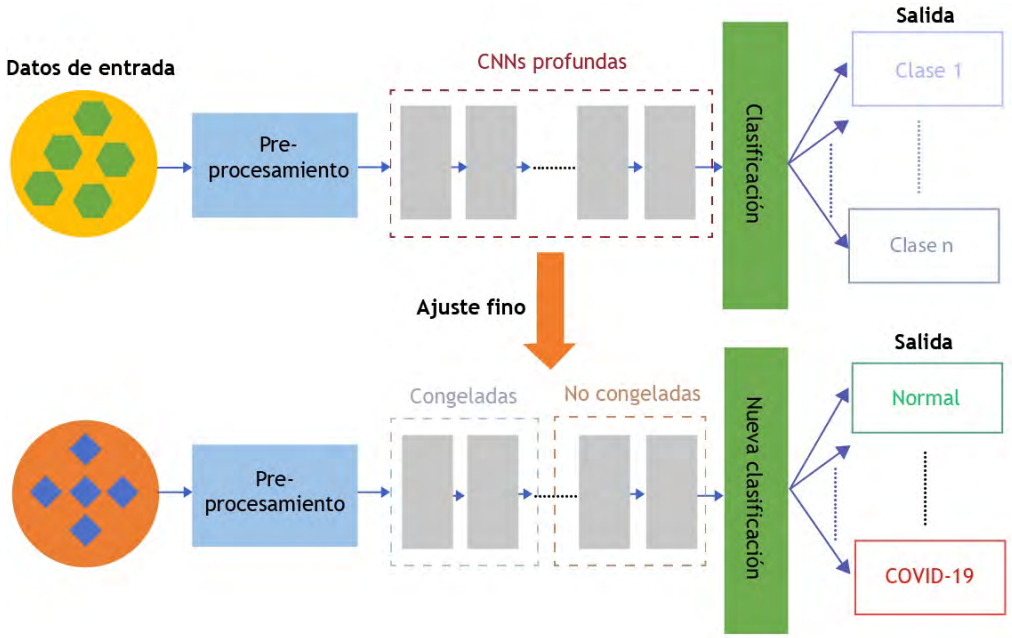
Nota. Adaptado de comprehensive review of deep learning-based methods for covid-19 detection using chest x-ray images (p. 100780), por S. Alahmari et al., 2022, IEEE Access, 10.1109. CC BY 4.0

Ajuste Fino (Fine-tuning). El ajuste fino es el proceso de utilizar conocimientos transferidos de un dominio diferente como pesos iniciales para entrenar un modelo de aprendizaje profundo para una tarea determinada. Este proceso aprovecha los modelos entrenados en grandes conjuntos de datos etiquetados para permitir el aprendizaje de modelos de forma eficaz para conjuntos de datos etiquetados más pequeños (Tajbakhsh *et al.*, 2016). Es decir, un modelo preentrenado en una tarea específica se utiliza como punto de partida y luego el conocimiento de este se adapta o ajusta finamente para abordar una nueva tarea (Véase la Figura 24).

En general, las primeras capas de una CNN aprenden características de imagen de bajo nivel, que son aplicables a la mayoría de las tareas de visión, pero las últimas capas

Figura 24

Esquema del ajuste fino.



Nota. Adaptado de comprehensive review of deep learning-based methods for covid-19 detection using chest x-ray images (p. 100780), por S. Alahmari et al., 2022, IEEE Access, 10.1109. CC BY 4.0

aprenden características de alto nivel, que son específicas de la aplicación en cuestión. Por lo tanto, el ajuste fino de las últimas capas suele ser suficiente para el aprendizaje por transferencia. Sin embargo, si la distancia entre las aplicaciones de origen y de destino es significativa, puede ser necesario ajustar también las primeras capas. Por lo tanto, una técnica eficaz de ajuste fino consiste en empezar por la última capa e ir incluyendo más capas en el proceso de actualización hasta alcanzar el rendimiento deseado. Denominamos "ajuste superficial" al ajuste de las últimas capas convolucionales y "ajuste profundo" al ajuste de todas las capas convolucionales(Tajbakhsh *et al.*, 2016).

Durante el ajuste fino, no solo se reemplaza el clasificador, sino que también se afinan los pesos de la red preentrenada mediante la continuación de la retropropagación. La cantidad de capas que se requiere ajustar en el proceso de ajuste fino depende de los datos utilizados y del tipo de red. Aunque es posible volver a entrenar todas las capas del modelo, es preferible mantener algunas capas anteriores congeladas para prevenir el sobreajuste, permitiendo el ajuste solo en algunas capas superiores de la red (Kc *et al.*, 2021).

Calidad de la Imagen

Según Wang *et al.* (2002), la mejor forma de evaluar la calidad de una imagen es mirarla, porque los ojos humanos son los receptores definitivos en la mayoría de los entornos de procesamiento de imágenes. Por otro lado, Engeldrum (2004) sugiere que la calidad de la imagen es la percepción integrada del grado general de excelencia de una imagen.

En contraste a los conceptos anteriores, según Thung y Raveendran (2009), históricamente, la calidad de imagen se describe en términos de visibilidad de las distorsiones de una imagen, como los cambios de color, el desenfoque, el ruido gaussiano y el efecto pixelado (blockiness). La forma más común de modelar una métrica de calidad de imagen es, por tanto, mediante la cuantificación de la visibilidad de estas distorsiones.

Y más recientemente, a efectos de la norma Organización Internacional de Estandarización (ISO) 19264 es necesario un enfoque ligeramente diferente relacionado con la reproducción de un original. El proceso de determinación de la calidad de la imagen implica una evaluación objetiva de la calidad de la imagen basada en una tabla de pruebas multipatrón que está diseñada para medir características de calidad de la imagen como el tono, el color y la reproducción de detalles, así como el ruido y la geometría (Wueller y Kejser, 2016).

Basándose en las definiciones y conceptos propuestos por los autores, se define a la calidad de imagen como “la percepción integral del grado de excelencia de las características de una imagen, expresada en términos de las distorsiones presentes en la misma”. Además, se reconoce que la mejor manera de evaluar la calidad de una imagen es a través de la observación visual, ya que los ojos humanos son los receptores finales de las imágenes.

Distorsiones de la Calidad de Imagen

Según la Real Academia Española (sf), la definición de la palabra distorsión es "Deformación de imágenes, sonidos, señales, etc., producida en su transmisión o reproducción". Por lo que en este contexto podríamos definir a la distorsión de la calidad de imagen como la alteración o deformación de las características de la imagen que dan por resultado la percepción de una imagen diferente por parte del espectador. A continuación en la Figura 25 se algunos ejemplos de distorsiones:

Figura 25

Evaluación de las imágenes de Lena distorsionadas por distintos medios

Nota. (a) Imagen original de Lena, 512×512, 8bits/píxel. (b) Imagen con el contraste aumentado. (c) Imagen contaminada por ruido gaussiano. (d) Imagen contaminada por ruido impulsivo. (e) Imagen borrosa (desenfoque). (f) Imagen con índice de compresión JPEG alto. Adaptado de Why is image quality assessment so difficult? (p. 4), por Z. Wang et al., 2002, IEEE International Conference on Acoustics, Speech, and Signal Processing. CC BY 4.0

Brillo y Contraste. El contraste y el brillo son dos características fundamentales que influyen en la percepción visual y la calidad de una imagen. Mientras que el brillo se

relaciona con el nivel global de luminosidad en una imagen, el contraste se refiere a las disparidades de luminosidad entre los elementos u objetos presentes en la imagen.

De acuerdo a Szeliski (2010) , en los primeros años de la televisión en blanco y negro, los materiales fosforescentes utilizados en los tubos de rayos catódicos para mostrar las señales de televisión respondían de manera no lineal a la tensión de entrada. Esta relación se expresa en la ecuación:

$$B = V^{\gamma} \tag{9}$$

Donde gamma (γ) con un valor cercano a $2{,}2$ representa la relación entre el voltaje y el brillo resultante. Para compensar el efecto de esta relación, los electrónicos de la cámara de la TV debían preajustar la luminancia detectada (Y) mediante una gamma inversa, y con un valor típico de $\frac{1}{\gamma} = 0{,}45$, como se describe en la siguiente fórmula:

$$Y\text{'} = Y^{\frac{1}{\gamma}} \tag{10}$$

Es de esta manera que mediante esta luminancia ajustada es posible ajustar el brillo de una imagen variando el parámetro (γ) dado que esta relación determina el nivel de luminosidad global de la imagen, en otras palabras el brillo. Por otro lado, Crane (1997) resalta que los ajustes de brillo y contraste son ejemplos de operadores puntuales dado que el valor de cada píxel de salida depende sólo del valor del píxel de entrada correspondiente (más, potencialmente, alguna información o parámetros recogidos globalmente).

Es así que Nixon y Aguado (2002) y Szeliski (2010) enuncian la siguiente expresión para determinar una nueva imagen con el brillo y el contraste ajustados:

$$g(x) = af(x) + b \tag{11}$$

Donde:

$g(x)$: Representa la nueva imagen.

$f(x)$: Representa la imagen de entrada.

$a > 0$: Controla el contraste.

b : Controla el brillo.

Es de esta manera que mediante esta expresión es posible modificar nivel del brillo y contraste de una imagen variando los parámetros a y b respectivamente.

Compresión JPEG. El algoritmo de compresión Joint Photographic Experts Group (JPEG) es un método ampliamente utilizado para comprimir imágenes digitales, especialmente fotografías y gráficos. Este formato de compresión se basa en la eliminación de datos redundantes o no perceptibles en una imagen con el fin de reducir su tamaño de archivo sin una pérdida significativa de calidad visual.

Según Gonzalez y Woods 2018 y Alcázar (2014), las etapas del algoritmo de compresión JPEG son las siguientes:

1. ***Transformación del espacio de color.*** El primer paso implica la modificación del espacio de color de la imagen. Normalmente, se emplea el espacio de color Rojo Verde Azul (RGB), que se transforma al espacio de color YCbCr (luminancia, crominancia azul y crominancia roja).

2. ***Submuestreo de crominancia.*** En esta etapa, se disminuye la resolución de los componentes de crominancia (Cb y Cr) en comparación con la luminancia (Y). Este proceso se realiza con el propósito de aprovechar la menor sensibilidad del ojo humano a las variaciones de color en contraste con su mayor sensibilidad a las variaciones de brillo.

3. ***Aplicación de la Transformada Discreta del Coseno (DCT).*** Este paso comienza dividiendo la imagen en bloques cuadrados de 8×8 píxeles. Luego, se aplica la DCT a cada bloque de píxeles. Esta transformación convierte los datos espaciales en datos de frecuencia, lo que permite eliminar información redundante en el dominio de la frecuencia.

4. ***Cuantización.*** En esta fase, los coeficientes derivados de la DCT se cuantifican, lo que implica redondearlos y almacenarlos con menor precisión. La relevancia de esta etapa radica en su capacidad para determinar el grado de compresión y la calidad de la imagen resultante. Esto se logra mediante la operación de los valores con una matriz de cuantización predefinida, según el nivel de compresión deseado. En la Figura 26 se muestra un ejemplo de una matriz de cuantización.

5. ***Codificación de longitud variable.*** Finalmente, después de cuantificar los valores, se lleva a cabo la codificación a nivel de bits utilizando una codificación de longitud variable, siendo el algoritmo Huffman el más comúnmente utilizado para representar los datos de forma eficiente.

Figura 26
Matriz de cuantización JPEG

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Nota. Reproducido de Digital Image Processing (Ed. n° 4, p. 589), R. C. Gonzalez y R. E. Woods, 2018, Pearson Education, Todos los derechos reservados 2018 por Pearson Education. Reproducido con permiso del autor.

Es así que para aplicar una distorsión JPEG basta con variar el nivel de compresión que consiste en escalar los valores de la matriz de cuantización que nombraremos como Z (Véase la Figura 26). Seguidamente, en la Figura 27 se pueden ver algunos ejemplos de la aplicación de diferentes matrices de cuantización.

Figura 27

Imagen comprimida con diferentes matrices de cuantización

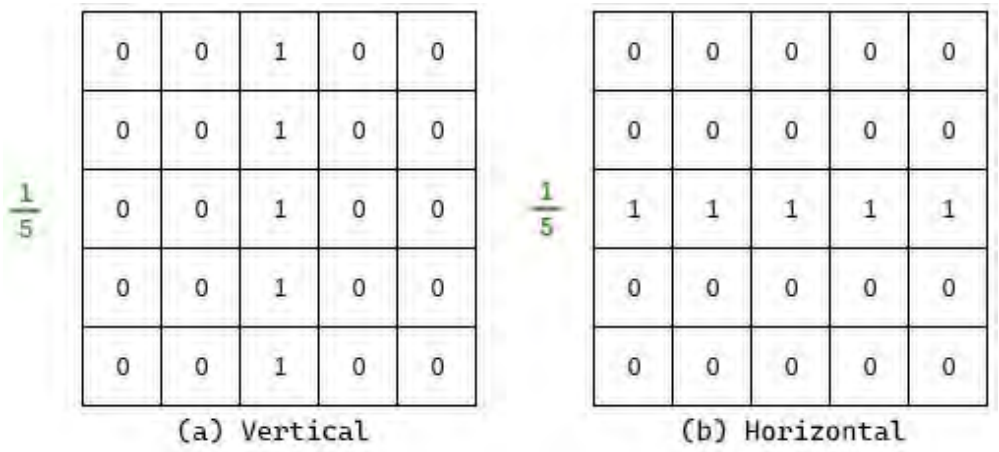
Nota. (a) Z, (b) 2Z, (c) 4Z, (d) 8Z, (e) 16Z, y (f) 32Z. Adaptado de Digital Image Processing (Ed. n° 4, p. 588), R. C. Gonzalez y R. E. Woods, 2018, Pearson Education, Todos los derechos reservados 2018 por Pearson Education. Adaptado con permiso del autor.

Desenfoque de Movimiento. El desenfoque de movimiento puede ser en forma de traslación, rotación y cambio repentino de la escala o algunas combinaciones de estas formas (Tiwari *et al.*, 2013). Esta investigación se centra principalmente en desenfoque por traslación, ya que es la forma más común de esta distorsión.

Esta distorsión se logra mediante la convolución de la imagen con un filtro normalizado de unos (Véase Figura 28). Cuanto mayor sea el tamaño del filtro, mayor será el efecto de desenfoque de movimiento. Además, la dirección de los unos en la cuadrícula del filtro es la dirección del movimiento deseado (Uberoi, 2019).

Figura 28

Filtros de desenfoque de movimiento de 5×5 *normalizado*



Nota. Adaptado de OpenCV | Motion Blur in Python, por GeeksforGeeks, Geeksforgeeks (https://www.geeksforgeeks.org/opencv-motion-blur-in-python/). Todos los derechos reservados 2023 por Sanchhaya Education Private Limited. Adaptado con permiso del autor.

Desenfoque Gaussiano. La distorsión del desenfoque o suavizado gaussiano se realiza mediante la convolución de la imagen con un kernel o filtro gaussiano.

Para la construcción del filtro gaussiano se parte de la función Gaussiana definida en una sola dimensión (Shapiro y StockMan, 2000).

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \tag{12}$$

Seguidamente para obtener la función necesaria para la construcción del kernel se multiplican dos funciones de una sola dimensión obteniendo la siguiente función de dos dimensiones (Reproducido de (Shapiro y StockMan, 2000)):

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{13}$$

Donde:

x : Distancia horizontal respecto del centro del filtro.

y : Distancia vertical respecto del centro del filtro.

σ : Desviación estándar.

Después de definir el filtro gaussiano utilizando la fórmula en la Ecuación 13, el siguiente paso implica normalizar los valores del filtro. Esta normalización es necesaria debido a que los valores obtenidos a través de la ecuación se distribuyen en un rango de tres desviaciones estándar (3σ), abarcando el 99.6 % de la distribución.

La normalización de los valores del filtro se lleva a cabo dividiendo cada valor entre la suma total de todos los valores en el filtro. A continuación, en la Figura 29, se muestra un ejemplo de un filtro normalizado.

Figura 29

Filtro gaussiano de 5×5 *normalizado*

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Nota. Adaptado de Computer Vision (p. 102), por R. Szeliski, 2010, Springer, Todos los derechos reservados 2011 por Springer-Verlag London. Adaptado con permiso del autor.

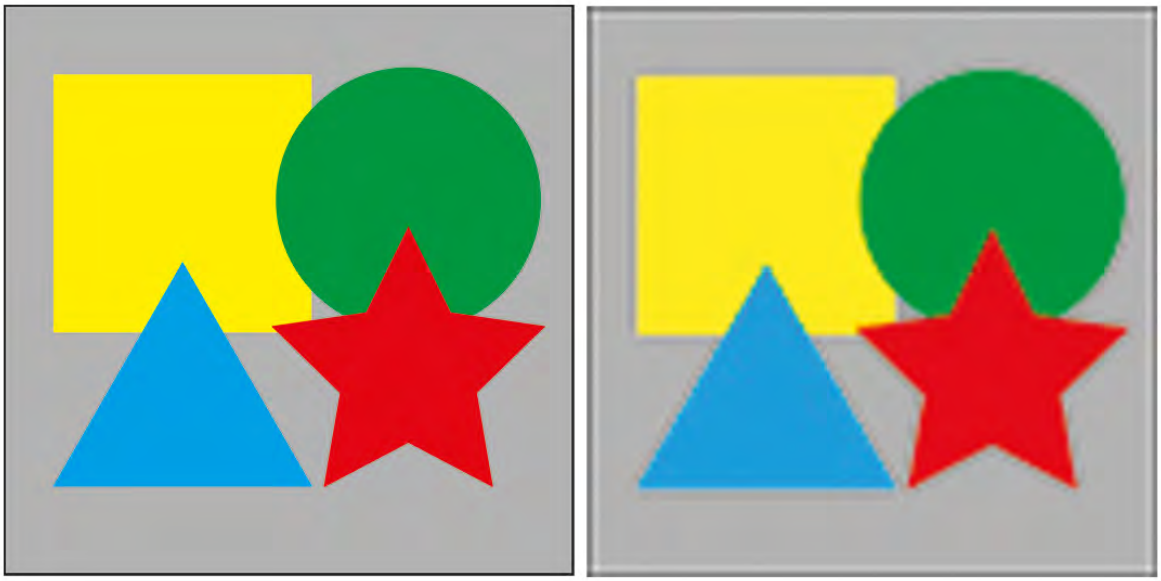
Resolución. La resolución de una imagen se refiere a la cantidad de detalles que se pueden ver en la imagen y se mide en términos de Píxeles por pulgada (PPP). La resolución determina la nitidez y la calidad de una imagen cuando se imprime o se muestra en una pantalla.

Para este proceso, se comienza por reducir la resolución de la imagen al tamaño de la resolución deseada y posteriormente se procede a ampliarla nuevamente a su tamaño original. Al disminuir el tamaño de la imagen y luego volver a aumentarlo, se está forzando una mayor densidad de píxeles en la misma área física, lo que a su vez significa una disminución en la resolución. Para llevar a cabo este proceso de reducción de la resolución, se utilizan algoritmos de redimensionado de imágenes, que pueden incluir algoritmos de interpolación así como algoritmos antialiasing más modernos.

En la Imagen 30 se puede ver un ejemplo del resultado de la reducción de resolución de 1080p a 108p por medio de interpolación bilineal.

Figura 30

Ejemplo de reducción de resolución de 1080p a 108p



Ruido Gaussiano. La introducción de esta distorsión se realiza mediante la generación de una matriz de ruido aleatorio que sigue una distribución normal lo que significa que los valores del ruido están distribuidos de una forma Gaussiana normal.

Según Gonzalez y Woods (2018), la función de densidad de probabilidad de una variable aleatoria gaussiana, z, se define mediante la conocida expresión:

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\bar{z})^2}{2\sigma^2}} \qquad -\infty < z < \infty \tag{14}$$

Donde:

z : Intensidad.

$\bar{z}$: Valor medio de z.

σ : La desviación estándar.

El ruido se agrega a la imagen original mediante la fórmula del modelo de ruido aditivo de la Ecuación 15.

$$w(x, y) = s(x, y) + n(x, y) \tag{15}$$

Donde:

x e y : Las coordenadas del píxel al que se aplica el ruido.

$w(x, y)$: La imagen distorsionada recibida tras aplicar el ruido.

$s(x, y)$: La intensidad de la imagen original.

$n(x, y)$: El ruido generado y añadido a la imagen original.

Desarrollo del Experimento

Este capítulo se basa en los pasos de la metodología propuesta (Véase la Figura 2), que comprende todos los aspectos de la captura de datos, la selección de las distorsiones de calidad de imagen, el procesamiento de imágenes, la construcción de datasets y finalmente la selección, implementación y entrenamiento de modelos CNN para la estimación del peso del cuy.

Captura de Datos

En esta fase, se realiza la captura de imágenes y el pesaje de los cuyes de forma simultánea. Antes de este proceso, la preparación del entorno es crucial, lo que implica aspectos como la adecuación de la iluminación, la disposición de la cámara y la configuración precisa de la balanza.

Preparación del Entorno

1. Para la iluminación, se seleccionan dos áreas o pozas de crianza disponibles en las granjas. El primer espacio se adapta utilizando la luz ambiental (Figura 31(a)) lo que significa que se utiliza la luz estándar que proviene de ventanas y tragaluces en el criadero. El segundo espacio se acondiciona con luz suplementaria (Figura 31(b)), concretamente, en un lugar donde la luz solar incide directamente en el suelo de forma perpendicular.

2. En cuanto a la cámara, se sigue una configuración estándar sin filtros y aditamentos, con la calidad del vídeo configurada a 1080p y 30FPS. Respecto al posicionamiento de la cámara se lleva a cabo un proceso en el que la cámara se sujeta a una altura aproximada de 1 metro, y se posiciona apuntando de forma perpendicular hacia la poza de crianza.

3. De manera similar, en lo que respecta a la configuración de la balanza digital, se coloca el recipiente destinado para el cuy y se utiliza la función de Tara (que permite restar el peso del recipiente para medir únicamente el contenido). Además,

se ajusta la balanza para mostrar los valores en libras (para obtener una mayor precisión), los cuales posteriormente se convierten a gramos.

Figura 31
Tipos de iluminación

(a) Luz ambiental (b) Luz suplementaria

Captura de Imágenes y Registro del Peso

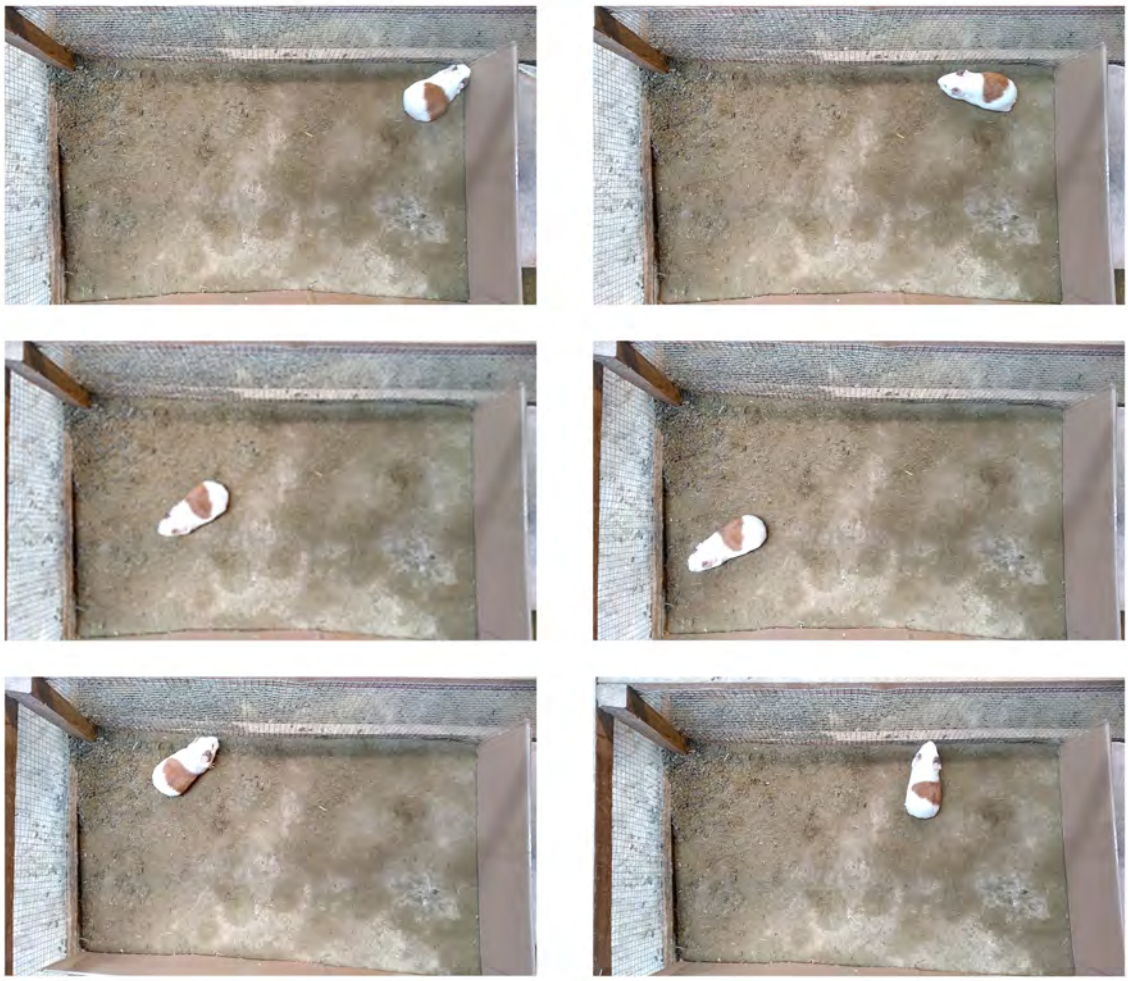
La captura de imágenes se realiza una vez se haya transportado el cuy hacia las pozas seleccionadas.

Las imágenes del cuy se obtienen grabando un vídeo con una duración que oscila entre 18 y 35 segundos. Se toma un plano rectangular para permitir una mayor movilidad y para capturar diversas posiciones del animal (Figura 32), asegurando que el cuerpo completo del cuy sea visible en el encuadre. Este proceso se repite de manera secuencial para las pozas de luz ambiental y suplementaria. Se registra el nombre del archivo de vídeo y luego se pasa al siguiente cuy en la poza correspondiente.

Luego de la captura de ambos vídeos, se procede a mover al cuy a una balanza digital para tomar registro de su peso. En total se capturaron un total de 118 vídeos para la luz ambiental y otros 118 para la luz suplementaria, lo que se traduce en que se tomó muestra de 118 cuyes.

Figura 32

Fotogramas de captura de imágenes



Selección de los Factores y Distorsiones de la Calidad de Imagen

La selección de los factores y distorsiones se realiza a partir de la revisión de trabajos de investigación existentes que tienen como tópico la evaluación de los efectos de la calidad de imagen sobre redes neuronales entrenadas para diversas tareas. Se elabora el Cuadro 5 a modo de resumen de los factores y distorsiones estudiadas en dichas investigaciones.

Cuadro 5

Factores y distorsiones evaluados en otras investigaciones

Distorsión / Factor	Ranjan *et al.* (2023)	Dodge y Karam (2016)	Jeelani *et al.* (2018)	Aqqa *et al.* (2019)	Hu *et al.* (2021)	Akkoca Gazioğlu y Kamaşak (2021)
Ruido gaussiano	X	X	X	-	X	X
Desenfoque gaussiano	X	X	-	-	X	X
Brillo	X	-	-	-	-	-
Contraste	-	X	-	-	-	X
Compresión JPEG	-	X	-	-	-	-
Resolución	-	-	-	-	X	-
Desenfoque de movimiento	-	-	-	-	X	-
Sobreexposición	X	-	-	-	X	-
Compresión JPEG2000	-	X	-	-	-	-
Saturación	X	-	-	-	-	-
Recorte	X	-	-	-	-	-
Mosaico	X	-	-	-	-	-
Iluminación	X	-	-	-	X	-
Tipo de sensor	X	-	-	-	-	-
Compresión de vídeo (CRF, bitrate)	-	-	-	X	-	-
Objetos externos	-	-	-	-	-	X
Aumento de datos	X	-	-	-	-	-
Tamaño del dataset	X	-	-	-	-	-
Función de pérdida (loss function)	-	-	X	-	-	-
Datasets diferentes e iguales para entrenamiento y prueba	-	-	-	-	X	-

Nota. Las filas sombreadas resaltan los factores y distorsiones seleccionadas.

Algunas distorsiones populares, como el ruido de sal y pimienta, no se incluyeron en este experimento, ya que, a diferencia de las distorsiones consideradas aquí, que afectan la estructura de la imagen de manera global, este tipo de ruido actúa de forma puntual, alterando únicamente un conjunto reducido de píxeles con valores extremos (blanco o negro). Además, debido a su naturaleza discreta y localizada, su impacto perceptual difiere significativamente del de las distorsiones que se distribuyen de manera uniforme en toda la imagen.

Esto es especialmente relevante en nuestro contexto de investigación, donde se simulan condiciones reales de captura en granjas, un entorno que no es ideal y presenta desafíos prácticos. En nuestro caso, se priorizan distorsiones como el desenfoque, el ruido gaussiano, la compresión JPEG y las variaciones de resolución, ya que son más representativas de los escenarios de degradación que se encuentran en la práctica. Por ejemplo, el desenfoque es común en imágenes capturadas en condiciones de movimiento o cuando la cámara no está bien enfocada. El ruido gaussiano puede deberse a errores

causados por el calor en el ambiente de las granjas o por condiciones de iluminación extremas (excesivas o deficientes). Por su parte, la compresión JPEG introduce artefactos característicos de este formato, que se usa por su capacidad de generar archivos de menor tamaño en comparación con otros algoritmos de compresión.

En cambio, el ruido de sal y pimienta es menos frecuente en escenarios reales y tiende a surgir por errores muy específicos de transmisión o defectos en los sensores, los cuales no siempre están presentes en la mayoría de las imágenes capturadas en condiciones cotidianas. Por ello, incluirlo junto a otras distorsiones podría desviar el análisis de cómo se comportan las imágenes bajo degradaciones más representativas en aplicaciones prácticas.

Factores de Calidad de la Imagen Seleccionados

El único factor seleccionado previo al inicio del experimento es la *iluminación* con el cual se realizará una evaluación preliminar e independiente.

Iluminación. Dado que la mayoría de las distorsiones se originan por fluctuaciones en los niveles de iluminación en los lugares donde se toman las imágenes, es necesario realizar un estudio independiente de la iluminación.

Los niveles de iluminación empleados en el experimento son los siguientes: “iluminación ambiental”, que corresponde a la luz natural que penetra en las áreas a través de ventanas y tragaluces, generando imágenes con sombras y contornos definidos por la extensión de la luz, e “iluminación suplementaria”, que comprende una fuente de luz natural o artificial que incide de manera directa y perpendicular sobre los objetos que se capturan en la imagen, anulando la presencia de sombras o reduciéndolas al mínimo.

Distorsiones de Calidad de la Imagen Seleccionadas

Estas son las siete distorsiones que se han seleccionado para su aplicación en las imágenes: *desenfoque gaussiano, desenfoque de movimiento, ruido gaussiano, brillo, contraste, compresión JPEG y resolución*. Las distorsiones seleccionadas son detalladas a continuación:

Desenfoque. Ciertos factores, como el movimiento de la cámara durante la toma de una fotografía, pueden resultar en una imagen borrosa, ya sea de manera deliberada, como en el caso de un barrido panorámico, o de manera no intencionada, debido a la falta de estabilidad en la sujeción de la cámara. Además, la elección de un tiempo de exposición breve en condiciones de poca luz o un enfoque incorrecto al capturar la imagen también puede conducir a la obtención de imágenes borrosas. Igualmente, la presencia de vibraciones o sacudidas durante la captura de la imagen, derivadas de factores ambientales o del manejo de la cámara, puede generar un efecto de desenfoque en la imagen resultante.

En base a estos factores mencionados que podrían ser comunes en las granjas en donde se pretende aplicar estos sistemas de estimación del peso animal se seleccionan las siguientes distorsiones:

Desenfoque gaussiano. Los niveles escogidos para esta distorsión se basan en la desviación estándar, denotada como "σ", que varía desde 1 hasta 10 en incrementos de 1. Se considera que $\sigma = 0$ representa la imagen original, $\sigma = 1$ indica el nivel más bajo de desenfoque gaussiano y $\sigma = 10$ representa el nivel con la mayor presencia de desenfoque gaussiano.

Desenfoque de movimiento. Para el desenfoque de movimiento, se seleccionan niveles que varían en función del tamaño del filtro o kernel, denotado por "ks". $ks = 0$ representa la imagen original, $ks = 3$ el nivel más bajo, y $ks = 12$ el nivel más alto de la distorsión, con intervalos de 1.

Ruido. Los sensores de imagen y cámaras pueden experimentar la influencia de diversos factores, incluyendo el calor generado por el propio sensor, las condiciones de iluminación, que pueden ser desde una insuficiente cantidad de luz hasta una luz excesiva, así como la no uniformidad de la cantidad de luz que llega a cada píxel, lo que conlleva a la introducción de variaciones en la señal capturada. Además, el ruido de lectura inherente al sensor, que provoca fluctuaciones y errores, junto con problemas en la conversión analógico-digital y posibles interferencias electromagnéticas y otras señales ambientales, pueden tener un impacto negativo en los dispositivos de captura de imágenes, dando lugar a

la aparición de ruido. Este ruido se refleja como píxeles brillantes u oscuros de manera aleatoria en la imagen. Por lo tanto, en vista de estos factores, se opta por abordar la siguiente distorsión:

Ruido gaussiano. Los niveles seleccionados de esta distorsión van determinados por una desviación estándar "σ" que varia desde 10 hasta 100 en intervalos de 10. Considerando que $\sigma = 0$ como la imagen original, $\sigma = 10$ el nivel más bajo y $\sigma = 100$ el nivel con más alta presencia de ruido gaussiano.

Brillo y Contraste. Diversos factores, como problemas de iluminación, ya sea por su escasez o exceso, representan causas primordiales de un desajuste en los niveles de brillo y contraste en una imagen. Cuando la escena que se está capturando carece de una iluminación adecuada, la cámara puede enfrentar dificultades para obtener una imagen equilibrada. Otro elemento determinante es la configuración incorrecta de la cámara en términos de exposición, que ocasionaría una inadecuada representación de brillo y contraste. Del mismo modo, los ajustes relacionados con la apertura, la velocidad de obturación y el valor ISO pueden tener un impacto negativo en estos aspectos. Además, es relevante tener en cuenta que algunas cámaras poseen limitaciones técnicas y, en situaciones específicas, puede resultar complicado capturar un amplio rango dinámico, que ocasionaría que el brillo y el contraste no sean los óptimos en dichas condiciones. Por tanto, es importante estudiar el brillo y el contraste.

Brillo. Los niveles elegidos dependen del parámetro "γ", que se encuentra en un rango de 0.2 a 4, con incrementos de 0.38. En este contexto, $\gamma = 1$ representa la imagen original. Los valores de γ en el rango de 0 a 1 indican niveles en los cuales el brillo oscurece la imagen, mientras que los valores en el rango de 1 a 4 representan niveles en los cuales el brillo aclara la imagen.

Contraste. Los niveles seleccionados varían en función del parámetro "α" que varia desde 0.2 hasta 4 en intervalos de 0.38. Considerando a $\alpha = 1$ como la imagen original, los valores $0 < \alpha < 1$ niveles en donde el contraste oscurece la imagen y los valores $1 < \alpha <= 4$ niveles en donde el contraste aclara la imagen.

Compresión y Resolución. Incrementar la compresión en un archivo JPEG puede llevar a obtener imágenes con un tamaño de archivo reducido, lo cual resulta beneficioso en situaciones donde se requiere economizar espacio en un disco o en dispositivos con capacidad de almacenamiento limitada. También facilita la rápida transmisión de imágenes, ya que el formato JPEG es conocido por su capacidad para comprimir imágenes al suprimir elementos redundantes o menos significativos, aunque a costa de sacrificar algunas de las características de la imagen.

Por otro lado, disminuir la resolución de una imagen genera también archivos de tamaño más reducido, lo cual puede ser ventajoso en circunstancias donde el espacio de almacenamiento es escaso. Sin embargo, esta reducción puede llevar a la pérdida de detalles esenciales de la imagen, por lo tanto es esencial realizar un estudio de esta distorsión.

Compresión JPEG. Los niveles seleccionados para esta distorsión se determinan a partir de una escala centesimal denominada "Nivel de calidad" y denotada por "lvl", que varía desde 2 hasta 20 en incrementos de 2. Se establece que $lvl >= 95$ corresponde a la imagen original, $lvl = 20$ denota el nivel más bajo de compresión gaussiana, y $lvl = 2$ indica el nivel con la máxima compresión. Se selecciona el nivel 20 como nivel mínimo de compresión porque a partir de dicho nivel recién se observan cambios significativos en la imagen.

Resolución. Los niveles de la resolución se denotan como "$resol$". En este contexto, se considera que $resol = 224p$ corresponde a la imagen original, $resol = 64p$ representa el nivel más bajo y $resol = 208p$ indica el nivel más alto de resolución evaluados, con incrementos de 16 entre ellos.

Procesamiento de Imágenes

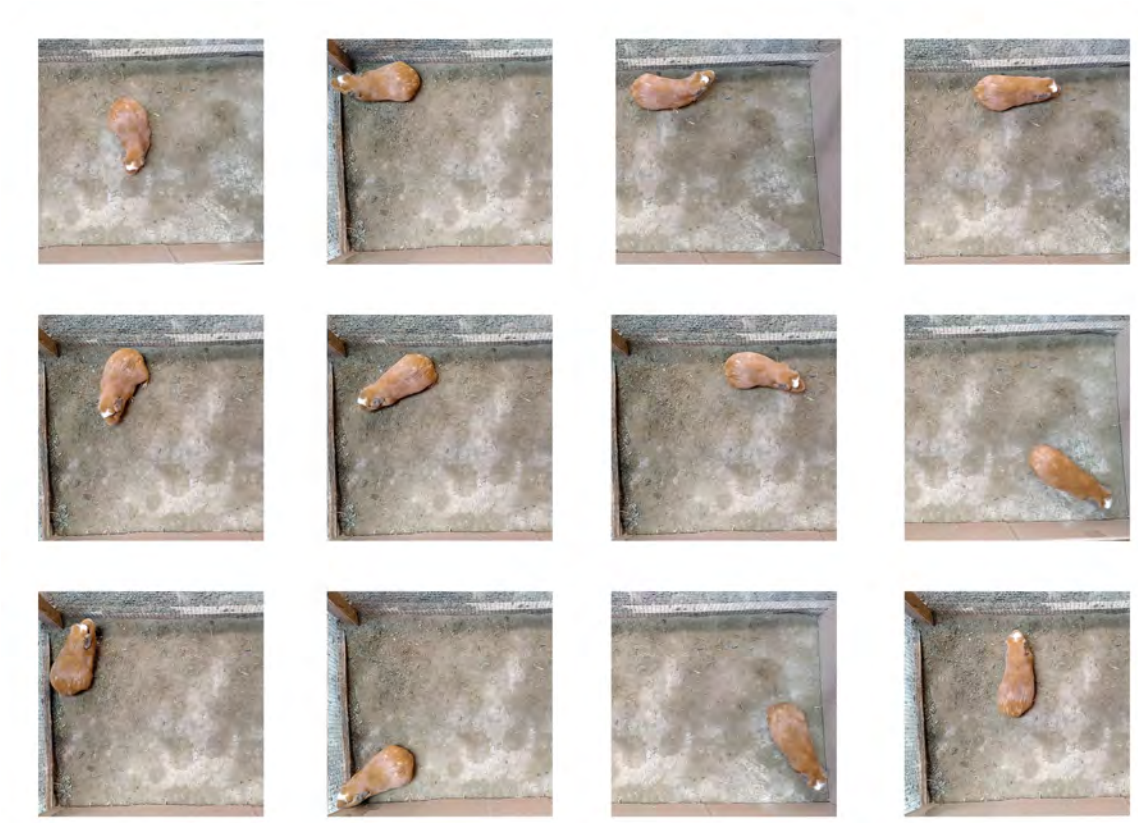
El procesamiento de imagen consta de dos etapas principales que se detallan a continuación:

Selección de Imágenes

El proceso de selección de imágenes comprende la extracción de fotogramas de los vídeos, el filtrado de fotogramas similares y su redimensión.

1. Los fotogramas se extraen mediante un script en Python que utiliza la librería OpenCV.
2. Se realiza una selección manual de los fotogramas similares, implicando una evaluación visual y juicio humano. Algunos de los criterios tomados en consideración son:
    - Se opta por imágenes en las cuales el cuerpo completo del cuy es visible.
    - El animal debe permanecer en una posición estática (sin efecto de movimiento).
    - Para filtrar las imágenes similares, se eligen las tomas en las que el animal ocupa posiciones distintas dentro del plano rectangular.
3. Finalmente, se seleccionan un aproximado de 13 imágenes de cada cuy (ver Figura 33) logrando agrupar un total de 1500 imágenes para el grupo de luz ambiental y otras 1500 para el grupo de luz suplementaria, se recorta cada imagen a un formato cuadrado de 1080×1080 píxeles, posteriormente se redimensionan a un tamaño final de 224×224 píxeles y, por último, se les asigna un nombre que incluye su identificador correspondiente.

Figura 33
Ejemplos de imágenes seleccionadas



Aplicación de las Distorsiones de Calidad y Aumento de Datos

Aumento de Datos. Inicialmente, el conjunto de datos se amplía mediante la rotación de las imágenes, generando nuevas imágenes a partir de las originales. En este estudio, optamos por aplicar únicamente rotación para evitar posibles interferencias con las distorsiones de calidad, que presentan características similares.

Para llevar a cabo este proceso, se utiliza el código presentado en el Código 1. Con este script, el objetivo es rotar las imágenes en ángulos de $90°$, $180°$ y $270°$ para evitar cortar el cuerpo del cuy, ya que uno de los requisitos previos es que en la imagen sea visible todo el cuerpo del animal. Como resultado, logramos generar un aumento en el número total de imágenes, cuadruplicándolo.

Listing 1
Script de rotación

```
from PIL import Image

# Leer imagen
img = Image.open(img_path)
# Rotación en grados
angle = 90
# Rotar la imagen
rotated_img = img.rotate(angle, expand=True)
```

Un ejemplo de los resultados del aumento de datos mediante la rotación se presenta a continuación en la Figura 34

Figura 34
Ejemplo de imagen aumentada

Aplicación de las Distorsiones. Luego del aumento de datos se procede a aplicar las distorsiones de calidad exclusivamente al conjunto de imágenes de prueba. Las distorsiones son aplicadas mediante un proceso automatizado de scripts en Python.

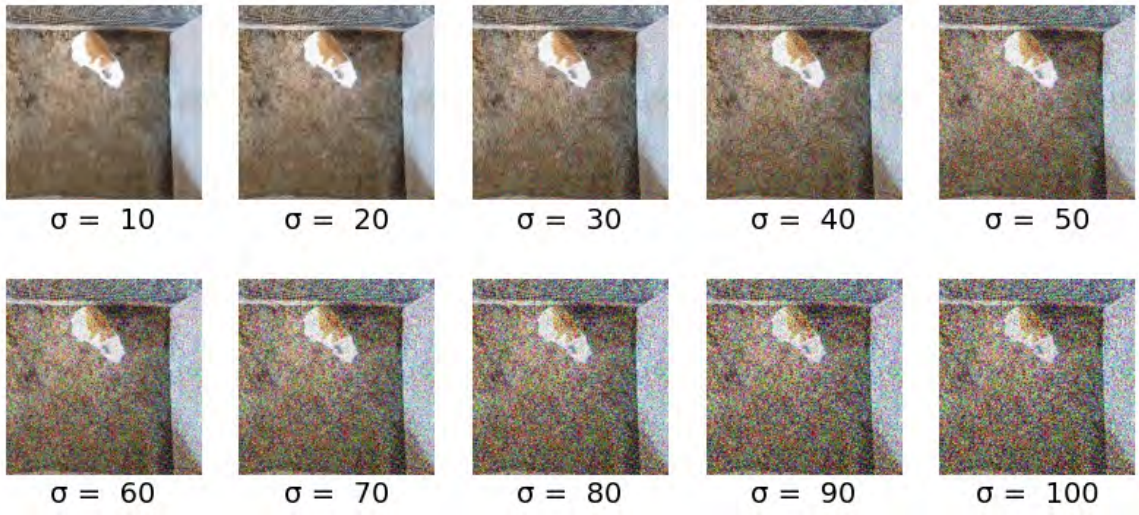
Ruido gaussiano. Como se observa en el Listing 2, la introducción de ruido gaussiano se realiza utilizando las librerías OpenCV y Numpy. La distorsión se introduce a través de una matriz de ruido generada con la función $randn$, que genera números aleatorios distribuidos normalmente con una media ($mean$) y una desviación estándar (std_dev). En este caso, el parámetro $mean$ se fija en 0, mientras que $std\_dev(\sigma)$ se ajusta para variar los niveles de distorsión.

Listing 2
Script de ruido gaussiano

```
import cv2
import numpy as np
# Leer imagen
img = cv2.imread(img_path)
# Crea una matriz de ruido gaussiano
mean = 0
std_dev = 80
noise = np.zeros(img.shape, np.int16)
# Se agrega el ruido aleatorio a cada canal por separado
cv2.randn(noise, (mean, mean, mean), (std_dev, std_dev, std_dev))
# Agrega el ruido a la imagen original
noisy_img = img + noise
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 35:

Figura 35
Imagen con distintos niveles de ruido gaussiano



Desenfoque gaussiano. Como se puede apreciar en el Código 3, la introducción de desenfoque gaussiano se realiza utilizando la biblioteca OpenCV.

La distorsión se incorpora mediante el módulo $GaussianBlur$, que recibe como parámetros una matriz de dimensiones $ker\_size \times ker\_size$ y una desviación estándar std_dev. En este contexto específico, el parámetro ker_size se calcula en función de std_dev con la fórmula propuesta por Akkoca Gazioğlu y Kamaşak (2021), donde se determina el ancho y alto de la matriz de la siguiente manera: $w, h = 2(2\sigma) + 1$. Sin embargo, para ajustarse al tamaño de las imágenes de 224×224 píxeles en nuestro estudio,

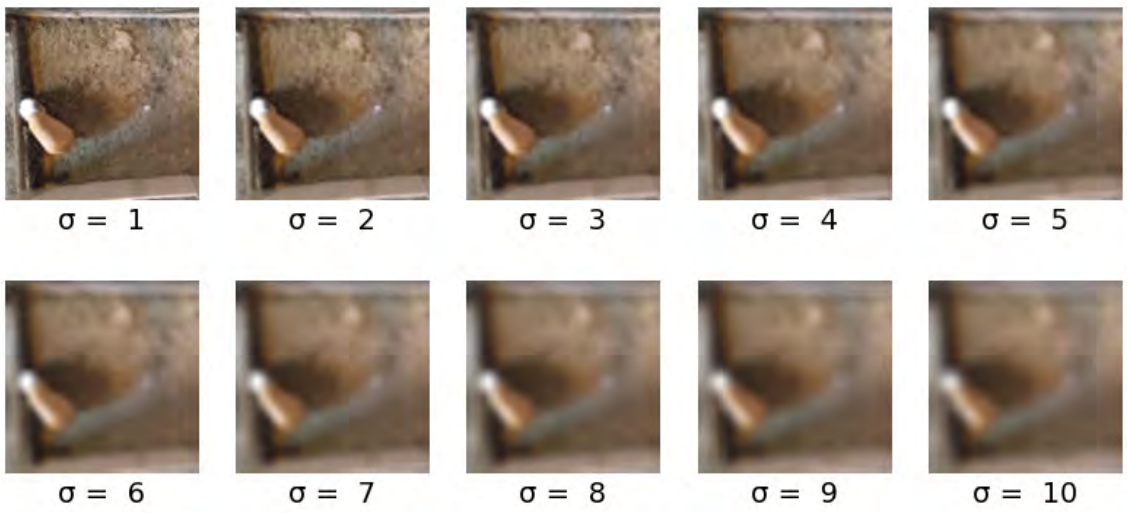
se establece $ker\_size = 2 * std\_dev + 1$. Por lo tanto, para variar el nivel de distorsión, se modifica únicamente el parámetro $std\_dev(\sigma)$.

Listing 3
Script de desenfoque gaussiano

```
import cv2
# Leer imagen
img = cv2.imread(img_path)
std_dev = 3 # Desviación estándar
ker_size = 2 * std_dev + 1  # Siempre debe ser impar
# Aplicar el desenfoque gaussiano
blurred_img = cv2.GaussianBlur(img, (ker_size, ker_size), std_dev)
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 36:

Figura 36
Imagen con distintos niveles de desenfoque gaussiano



Contraste. En el Listing 4 se muestra la implementación de la distorsión de contraste utilizando la biblioteca OpenCV.

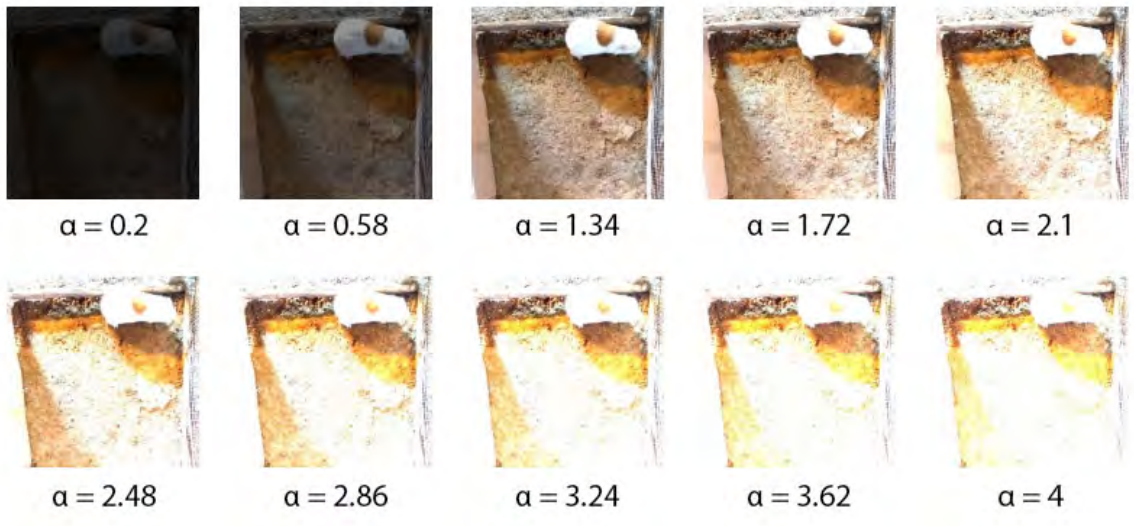
La distorsión se aplica utilizando el módulo $convertScaleAbs$, manteniendo el factor $beta$ en 0 y ajustando el parámetro $alpha(\alpha)$ para controlar los niveles de contraste.

Listing 4
Script de contraste

```
import cv2
# Leer imagen
img = cv2.imread(img_path)
# Variar el factor alpha
alpha = 2.1 # Factor de contraste
beta = 0
# Ajustar el contraste
adjusted_img = cv2.convertScaleAbs(img, alpha=alpha, beta=beta)
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 37:

Figura 37
Imagen con distintos niveles de contraste



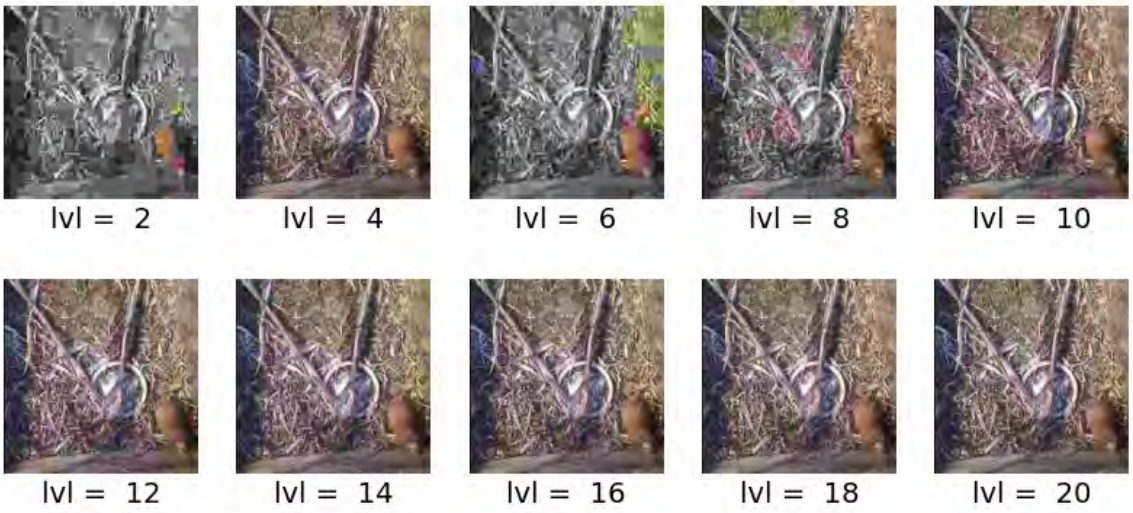
Compresión JPEG. Como se observa en el Listing 5, la compresión JPEG se ajusta mediante la librería Pillow, que permite especificar el nivel de compresión con el parámetro *lvl*.

Listing 5
Script de compresión JPEG

```
from PIL import Image
# Leer imagen
img = Image.open(img_path)
# Factor de calidad JPEG
lvl = 2
# Guardar la imagen con un nivel JPEG determinado
img.save(new_img_route, quality=lvl)
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 38:

Figura 38
Imagen con distintos niveles de compresión JPEG



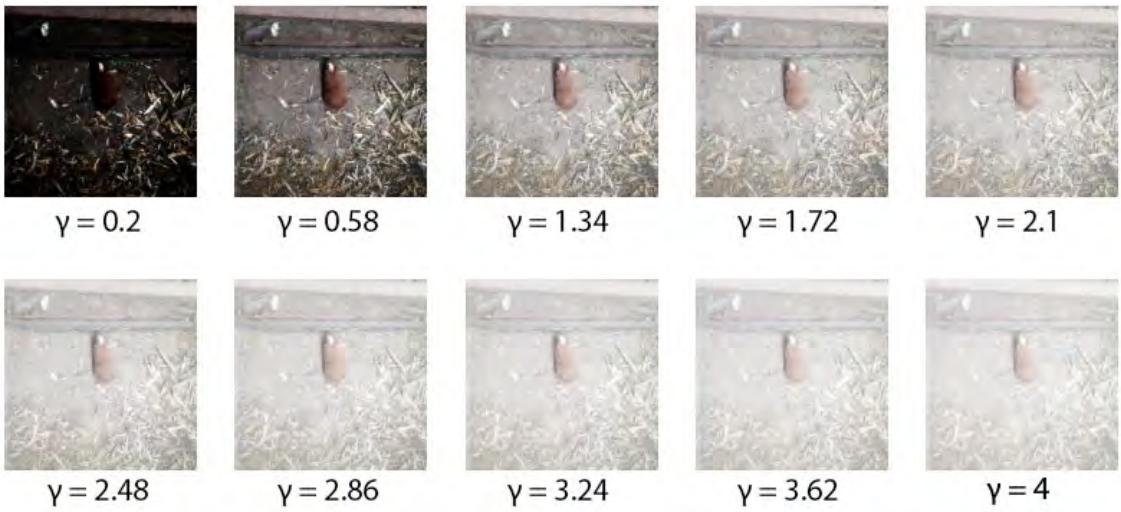
Brillo. En el Listing 6, se presenta la implementación de la distorsión de brillo utilizando las bibliotecas Numpy y OpenCV. La distorsión se realiza utilizando el módulo $adjust\_gamma$, que invierte el factor $gamma(\gamma)$ para crear una matriz de ajustes para los píxeles. Esta matriz se utiliza para mapear los valores de los píxeles de la imagen a nuevos valores. Esta operación se lleva a cabo utilizando el módulo LUT de OpenCV. En general, para modificar la imagen, basta con ajustar el valor de la variable $gamma(\gamma)$.

Listing 6
Script de brillo

```
import cv2
import numpy as np
def adjust_gamma(image, gamma=1.0):
    inv_gamma = 1.0 / gamma
    table = np.array([((i / 255.0) ** inv_gamma) * 255
       for i in np.arange(0, 256)]).astype("uint8")
    return cv2.LUT(image, table)
# Leer imagen
img = cv2.imread(img_path)
# Factor de brillo
gamma=3.62
# Ajustar el brillo de la imagen
adjusted_img = adjust_gamma(img, gamma=gamma)
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 39:

Figura 39
Imagen con distintos niveles de brillo



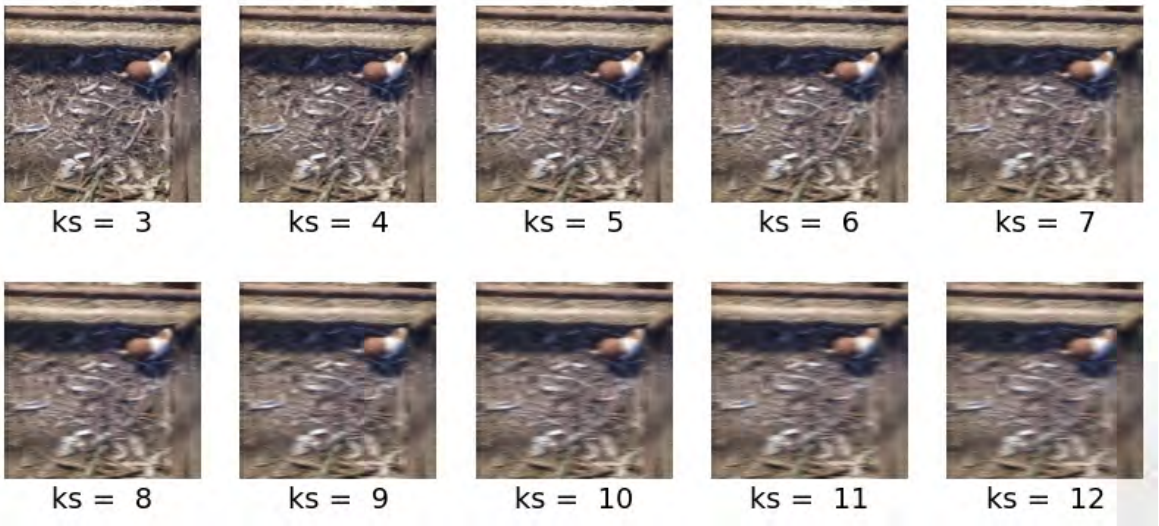
Desenfoque de movimiento. Como se puede observar en el Listing 7, la aplicación del desenfoque de movimiento se efectúa mediante la convolución con una matriz que contiene ceros, excepto por una fila de unos en el centro (lo que produce un desenfoque horizontal) que posteriormente se normaliza de acuerdo al tamaño de la matriz. La creación de esta matriz se lleva a cabo utilizando la biblioteca Numpy, y la convolución o aplicación del filtro se realiza mediante el módulo $filter2D$ de OpenCV. Las variaciones en el nivel de distorsión están determinadas por el tamaño de la matriz que se utiliza, en este caso, el valor de $kernel\_size(ks)$.

Listing 7
Script de desenfoque de movimiento

```
import cv2
import numpy as np
# Leer imagen
img = cv2.imread(img_path)
kernel_size = 12
# Crear el kernel
kernel = np.zeros((kernel_size, kernel_size))
# Rellenar la fila del medio con unos (horizontal)
kernel[int((kernel_size - 1)/2), :] = np.ones(kernel_size)
# Normalizar
kernel /= kernel_size
# Aplicar el filtro
image_with_mb = cv2.filter2D(img, -1, kernel)
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 40:

Figura 40
Imagen con distintos niveles de desenfoque de movimiento



Resolución. La distorsión relacionada con la resolución se implementa utilizando el código presentado en el Código 8. La variación de la resolución se logra mediante el módulo $resize$ de la biblioteca OpenCV, que permite redimensionar la imagen a un tamaño más pequeño. Esta operación se realiza con la intención de, posteriormente, volver a redimensionar la imagen a un tamaño más grande, lo que altera la resolución de la imagen. Para ajustar la resolución, basta con modificar el valor del parámetro $resol$.

Listing 8

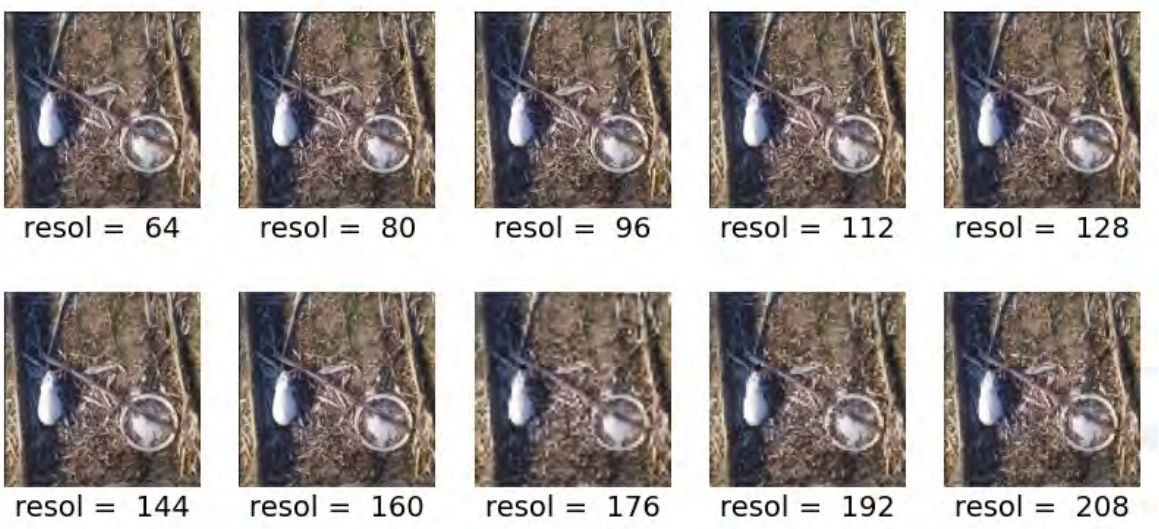
Script de resolución

```
import cv2
# Leer imagen
img = cv2.imread(img_path)
resol = 96 # Resolución deseada
# Guardar imagen con menor tamaño
img_resized = cv2.resize(img, (resol, resol))  # Redimensionar
cv2.imwrite('aux.jpg', imgResized)  # Guardar imagen temporal
# Volver a redimensionar imagen al tamaño original, pero con
# menor resolución
img2 = cv2.imread('aux.jpg')  # Leer imagen temporal
img_resized2 = cv2.resize(img2, (224, 224))  # Redimensionar
```

Los resultados de la aplicación de esta distorsión se pueden ver en la Figura 41:

Figura 41

Imagen con distintos niveles de resolución



Construcción de Datasets

El proceso de construcción de los datasets consiste en agrupar las imágenes junto con sus respectivos pesos. Luego, cada dataset se divide en una proporción de 80:10:10 para entrenamiento, validación y prueba, respectivamente. La descripción del dataset principal se puede apreciar en el Anexo B. A continuación, se describe el método utilizado para la división de todos los datasets.

1. Se ordena la lista de cuyes y sus pesos en orden ascendente según los pesos.
2. Se divide el conjunto total de datos en deciles.

3. Se toman dos o tres cuyes de cada decil. Dependiendo del número de imágenes, estas se agregan al grupo de validación o de prueba, hasta que se alcancen la proporciones planificadas.

Estas medidas aseguran que en los grupos de entrenamiento, validación y prueba exista una proporción equilibrada de imágenes de cuyes con pesos similares, y que las imágenes del mismo cuy no se utilicen tanto para entrenar como para probar los modelos, garantizando así la calidad de los conjuntos de datos.

Datasets de Iluminación

Para llevar a cabo las pruebas de iluminación, se crean datasets para los dos tipos de iluminación: luz ambiental y luz suplementaria, y un dataset adicional con una combinación de imágenes de ambos. Con el objetivo de mantener la imparcialidad en la cantidad de datos, los tres datasets constan de un total de 6000 imágenes cada uno, generadas a partir de los 1500 fotogramas originales a los que solo se les aplicó el aumento de datos. A continuación, se describe la estructura de estos datasets:

- **luz_ambiental / luz_suplementaria / combinado**
    - **train** (carpeta con 4800 imágenes de entrenamiento)
    - **val** (carpeta con un aproximado de 600 imágenes de validación)
    - **test** (carpeta con un aproximado de 600 imágenes de prueba)
    - **pesos.csv** (archivo CSV con los 6000 registros de nombres de las imágenes y sus respectivos pesos)

Además es importante aclarar que el dataset combinado contiene una misma proporción de imágenes (50 % - 50 %) de los datasets $luz\_ambiental$ y $luz\_suplementaria$. Este enfoque busca asegurar que los tres datasets sean comparables y que la evaluación de los modelos se realice de manera justa bajo diferentes condiciones de iluminación.

Dataset Principal

Este dataset se construye en base a los resultados obtenidos de la experimentación con los datasets de iluminación. Incluye todas las imágenes del dataset de luz ambiental y luz suplementaria, y adicionalmente se agregan las imágenes del dataset del trabajo de Zapata-Ttito (2022). En total, este dataset principal contiene 4561 imágenes, que se someten a aumentos de datos, resultando en un conjunto final de 18244 imágenes. La estructura de este dataset principal después de la distribución se compone de la siguiente manera:

- **principal**
    - **train** (carpeta con las 14596 imágenes de entrenamiento)
    - **val** (carpeta con las 1828 imágenes de validación)
    - **test** (carpeta con las 1820 imágenes de prueba)
    - **pesos.csv** (archivo CSV con los 18244 registros de nombres de las imágenes y sus respectivos pesos)

Este dataset principal es una compilación de diferentes fuentes de imágenes y es utilizado para realizar los experimentos y evaluaciones base de los modelos de estimación de peso animal.

Datasets de Distorsiones

Para realizar las evaluaciones de las distorsiones, se emplea el conjunto de imágenes de prueba del dataset principal como punto de partida. A partir de este conjunto, se generan 10 conjuntos de prueba para cada una de las siete distorsiones seleccionadas: ruido gaussiano, desenfoque gaussiano, contraste, compresión JPEG, brillo, desenfoque de movimiento y resolución. A continuación, se detalla la organización de cada dataset de acuerdo a cada distorsión y los niveles para cada una en el Cuadro 6:

- **desenfoque_gauss / desenfoque_mov / ruido_gauss / contraste / brillo / jpeg / resolucion**

- **test** (carpeta con las 1820 imágenes de prueba distorsionadas)
- **pesos.csv** (archivo CSV con los 1820 registros de nombres de las imágenes y sus respectivos pesos)

Cuadro 6
Distorsiones, parámetros y niveles

Distorsión	Parámetro	Niveles									
Desenfoque gaussiano	σ	1	2	3	4	5	6	7	8	9	10
Desenfoque de movimiento	ks	3	4	5	6	7	8	9	10	11	12
Ruido gaussiano	σ	10	20	30	40	50	60	70	80	90	100
Brillo	γ	0.2	0.58	1.34	1.72	2.1	2.48	2.86	3.24	3.62	4
Contraste	α	0.2	0.58	1.34	1.72	2.1	2.48	2.86	3.24	3.62	4
Compresión JPEG	lvl	20	18	16	14	12	10	8	6	4	2
Resolución	$resol$	208	192	176	160	144	128	112	96	80	64

Cada conjunto de prueba se crea a partir de imágenes del conjunto de prueba original del dataset principal y se aplica una única distorsión específica a las imágenes seleccionadas para ese conjunto. No se aplican aumentos de datos adicionales a estos conjuntos de prueba, lo que permite evaluar el impacto de cada distorsión individualmente en el rendimiento de los modelos de estimación del peso animal.

Selección, Implementación y Entrenamiento de Modelos de CNN

Selección de los Modelos

La exploración de arquitecturas de CNN ha sido fundamental en el estudio de redes neuronales desde su descubrimiento inicial. El auge actual de las redes neuronales ha revitalizado este campo de investigación. El aumento en el número de capas y las diversas propuestas en la arquitectura de las redes contemporáneas intensifica las diferencias entre estas estructuras, estimulando la exploración de diferentes patrones de conexión y la reconsideración de antiguas ideas de investigación.

La elección de modelos se basa en la investigación de Akkoca Gazioğlu y Kamaşak (2021), donde se busca aplicar la misma técnica utilizada para adaptar los modelos a su problemática específica. En su trabajo, los autores optaron por los modelos ResNet50, DenseNet121 y VGG16.

Sin embargo, se modifica la elección de la arquitectura VGG16 debido a un análisis

comparativo de arquitecturas realizado en el trabajo de Bianco *et al.* (2018) (consúltese el Cuadro 7). Según este análisis, para lotes de imágenes de 32, los modelos basados en las arquitecturas VGG (Simonyan y Zisserman, 2015) tienen un consumo de memoria superior a 3.5GB, en contraste con la red ResNet50 que consume 1.28GB y la DenseNet121 con 0.71GB. Esto se atribuye a que las redes VGG siguen la filosofía de construir redes "planas" mediante el apilamiento de capas (He *et al.*, 2015), lo que nos lleva a reconsiderar el uso de esta arquitectura.

Cuadro 7
Memoria utilizada por las CNN propuestas

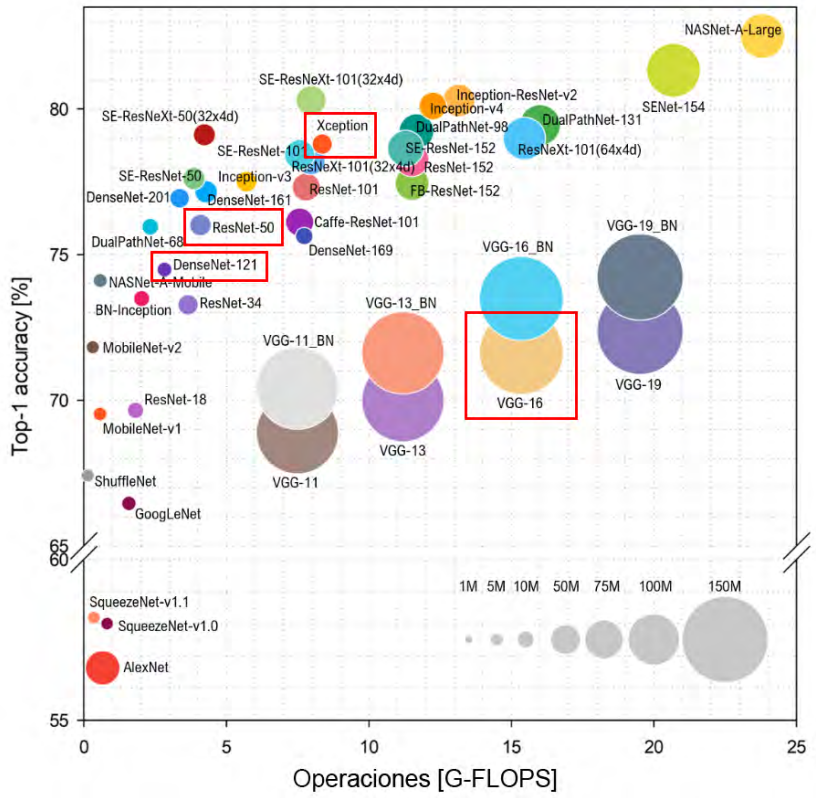
Modelo de CNN	Memoria usada (GB)
ResNet50	1.28
DenseNet121	0.71
VGG16	3.61
Xception	1.24

Nota. Adaptado de Benchmark Analysis of Representative Deep Neural Network Architectures (p. 7), por S. Bianco, 2018, arXiv, 1810.00736v2. CC BY 4.0

En lugar de optar por VGG16, se decide seleccionar la red Xception como alternativa, ya que, según su creador (Chollet, 2017), esta guarda similitudes esquemáticas con VGG16 en ciertos aspectos. Además, en comparación con las otras dos arquitecturas, Xception presenta un consumo de memoria similar a ResNet50 (véase el Cuadro 7) y exhibe un rendimiento superior en tareas de clasificación en comparación con los otros dos modelos. Alcanza un puntaje de Top-1 Accuracy cercano al 79 %, en contraste con el 76 % de ResNet50, el 75.4 % de DenseNet121 y el casi 72 % de VGG16 (véase la Figura 42). Por lo tanto, surge el interés de explorar si esta red constituye una opción viable para las tareas de regresión.

Figura 42

Top-1 Accuracy vs complejidad computacional (FLOPs)



Nota. Adaptado de Benchmark Analysis of Representative Deep Neural Network Architectures (p. 3), por S. Bianco, 2018, arXiv, 1810.00736v2. CC BY 4.0

La elección de arquitecturas clásicas como ResNet, DenseNet y Xception en lugar de modelos modernos se fundamenta en su trayectoria probada y consolidada en una amplia gama de tareas, lo que brinda mayor confianza en sus resultados. Estas arquitecturas han sido evaluadas en profundidad y optimizadas a través de múltiples estudios, lo que asegura su robustez frente a diferentes escenarios de uso. En cambio, las redes más recientes como EfficientNet o NASNet, aunque prometen mejoras en eficiencia y rendimiento, aún requieren una mayor exploración y validación en tareas específicas para entender completamente sus limitaciones y adaptabilidad.

Las arquitecturas modernas tienden a ser más especializadas y optimizadas para entornos de vanguardia, pero carecen de la extensiva evaluación y refinamiento que las arquitecturas clásicas han recibido a lo largo de los años. Esto implica que en aplicaciones con condiciones no estándar o datos de baja calidad, como la estimación de peso en

imágenes de cuyes, las arquitecturas clásicas proporcionan un rendimiento más predecible y estable. Por lo tanto, las redes neuronales establecidas representan una opción más confiable cuando la estabilidad y consistencia de los resultados son factores cruciales.

Implementación de los Modelos

Para implementar los modelos elegidos, se utiliza "Colaboratory", un entorno de desarrollo de Python basado en la nube, enfocado en la programación colaborativa y la ejecución de proyectos de Aprendizaje Automático.

Para habilitar a las arquitecturas seleccionadas a estimar el peso de los cuyes, se procede con la implementación siguiendo un conjunto de pasos idénticos para cada una de las tres arquitecturas: ResNet50, DenseNet121 y Xception. Los siguientes pasos son importantes para configurar adecuadamente cada modelo con los hiperparámetros óptimos determinados en el proceso de entrenamiento de los modelos en la siguiente sección.

1. **Inicialización del dataset de imágenes.** Inicialmente, se carga el conjunto de imágenes y pesos utilizando un "Image Data Generator" con la ayuda de las bibliotecas "Tensorflow" y "Pandas". Dentro de este generador de datos, los píxeles de las imágenes se normalizan para que estén en un rango de 0 a 1, lo que mejora el rendimiento durante el entrenamiento evitando el desvanecimiento de gradiente. Además, se establece el tamaño de los lotes de imágenes en 32 y se habilita la mezcla aleatoria de imágenes para evitar que el modelo aprenda únicamente patrones específicos del orden de las imágenes.

2. **Implementación del modelo base y transferencia de aprendizaje.** A continuación, se implementa el "modelo base" utilizando Tensorflow y Keras. Dado que Tensorflow proporciona las arquitecturas de las tres redes: ResNet50, DenseNet121 y Xception, estas se inicializan con los pesos preentrenados de "Imagenet" y sin la última capa totalmente interconectada (normalmente utilizada para clasificación).

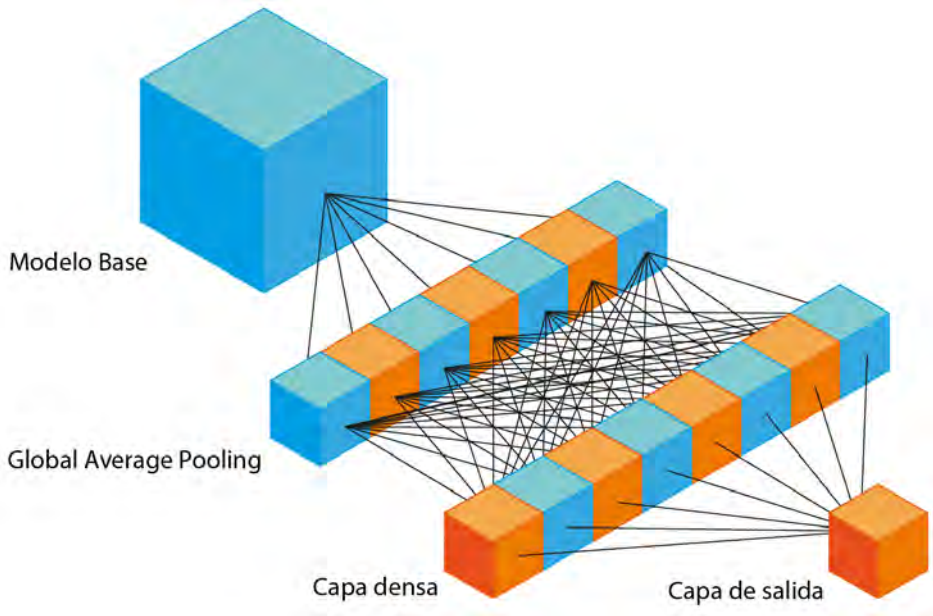
3. **Definición del tamaño de imagen.** Se configura el tamaño de entrada de las

imágenes a 224×224 píxeles con tres canales de colores (RGB).

4. **Implementación del regresor de peso.** Este estimador consta de nuevas capas, que incluyen una capa "Global Average Pooling 2D", una capa densa con 512 neuronas y una función de activación ReLU, y finalmente una capa de salida densa con una sola neurona que se utiliza para estimar el peso. La función de activación de esta última capa también es una función ReLU (Véase Figura 43). Luego estas capas son enlazadas al modelo base.

5. **Compilación y definición de métricas.** Finalmente, después de combinar el modelo base con el estimador personalizado, se compilan los modelos utilizando el optimizador Adam (Kingma y Ba, 2017). Se define la función de pérdida como el Error Cuadrático Medio (MSE) y se utilizan métricas de diagnóstico como el Error Absoluto Medio (MAE) y el Error Absoluto Porcentual Medio (MAPE). Luego de estos pasos, las redes quedan preparadas para el proceso de entrenamiento.

Figura 43
Diagrama del regresor de peso



Entrenamiento de los Modelos

Ajuste de Hiperparámetros. Previo al entrenamiento de los modelos, se lleva a cabo el ajuste de hiperparámetros mediante varias experimentaciones con el conjunto de

validación, tal como se detalla a continuación:

- ***Tamaño de imagen.*** Se refiere a las dimensiones (alto x ancho) de las imágenes que se ingresan a la red. Este tamaño influye en el uso de memoria y el rendimiento del modelo. Generalmente, se usan tamaños como 224×224, 256×256 y 128×128 píxeles. Elegir el tamaño adecuado es crucial para capturar suficiente información sin sobrecargar la capacidad de procesamiento de la GPU.

  Se selecciona el tamaño 224×224 *porque se utilizarán modelos preentrenados con estas dimensiones de imagen.*

- ***Tamaño de lotes de imagenes (batch size).*** Este parámetro indica el número de muestras de datos que se procesan antes de actualizar los parámetros del modelo.

  Se establece en 32, considerando el tamaño de la imagen de entrada, la memoria del procesador gráfico y el tiempo de entrenamiento, según se observa en el Cuadro 8

Cuadro 8
Observaciones del tamaño de lotes

Batch size	Observaciones
16	• No realiza un uso completo de la memoria de la GPU. • El entrenamiento es ligeramente más rápido en comparación a 32. • El MAPE del conjunto de validación de los modelos supera el 30 %.
32	◦ Realiza uso casi del total de la Memoria de Acceso Aleatorio de Video (VRAM) de la GPU. ◦ El tiempo de entrenamiento es aceptable al compararlo con el batch size de 16. ◦ El MAPE del conjunto de validación de los modelos es inferior a 30 %.
> 32	⋆ Supera el uso de VRAM de la GPU, por lo que no es posible continuar con el entrenamiento.

- ***Función Early Stopping.*** La función de detención anticipada (Early Stopping) se implementa para prevenir el sobreajuste. Esta estrategia detiene el entrenamiento si se observa un aumento en la función de pérdida (MSE) del conjunto de validación después de un cierto número de épocas.

La detención anticipada se establece con una tolerancia de 5 épocas.

- ***Número de épocas.*** Define cuántas veces se utiliza todo el conjunto de datos de entrenamiento para actualizar los pesos del modelo.

  Inicialmente, este parámetro se fija en 35, y luego los modelos se vuelven a entrenar hasta alcanzar 50 y 80 épocas. No obstante, se observa que las curvas de entrenamiento convergen en etapas tempranas y los valores óptimos se encuentran alrededor de la época 35. Por lo tanto, se considera que 35 épocas es el valor óptimo.

- ***Optimizador.*** Es el algoritmo que ajusta los pesos y sesgos de la red durante el entrenamiento para minimizar la función de pérdida.

  Se elige el optimizador Adam basándose en los resultados obtenidos en el trabajo de Chen et al. (2023).

- ***Tasa de aprendizaje.*** Este hiperparámetro determina el tamaño de los pasos que da el optimizador al actualizar los pesos y sesgos durante el entrenamiento. Elegir un valor adecuado es crucial para lograr una mejor convergencia y velocidad de entrenamiento.

  La optimización de la tasa de aprendizaje de los modelos se realiza basado en los tipos de iluminación y el optimizador Adam. Por lo tanto, este hiperparámetro se calcula en base a la Ecuación 16 determinada en el trabajo de Chen et al. (2023):

$$lr = lr_{inicial} \times dr^{\frac{gs}{ds}} \tag{16}$$

Donde:

lr : Representa a la tasa de aprendizaje calculada.

$lr_{inicial}$: Denota la tasa de aprendizaje inicial. Inicialmente establecida en 10^{-5} y posteriormente ajustada a los modelos en específico.

dr : Representa la tasa de decaimiento y definida en 0.99.

ds : Denota el número de etapas de decaimiento. Definida inicialmente en 1000, seguidamente ajustada a cada modelo.

gs : Representa el número global de etapas del modelo. En nuestro caso es lo mismo que el "número de pasos por época".

La optimización de la tasa de aprendizaje de los modelos se realiza basados en la luz ambiental, luz suplementaria y un dataset combinado. En este contexto, la mayoría de los valores sugeridos por Chen *et al.* (2023) se mantuvieron en sus configuraciones iniciales, excepto la tasa de aprendizaje inicial ($lr_{inicial}$). A continuación, en los Cuadros 9, 10, 11 se proporcionan los resultados obtenidos con los datos de prueba de cada conjunto de datos para cada uno de los modelos seleccionados.

Los valores óptimos para la tasa de aprendizaje inicial ($lr_{inicial}$) para ResNet50, DenseNet121 y Xception son $10^{-4,7}$*,* $10^{-4,3}$ *y* $10^{-4,2}$*, respectivamente.*

Cuadro 9

Pruebas de variación de la tasa de aprendizaje inicial de ResNet50.

		ResNet50							
$lr_{inicial}$	**Épocas**	**Luz baja**		**Luz Alta**		**Combinado**		**Promedio**	
		MAE	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**
$10^{-4,8}$	50	65.59	14.51	69.31	13.47	70.68	13.62	68.52	13.87
$10^{-4,7}$	35	58.86	13.32	66.54	12.31	68.97	13.66	64.79	13.10
$10^{-4,6}$	35	62.21	12.60	68.40	13.55	70.24	14.33	66.95	13.49

Nota. La fila sombreada resalta el parámetro seleccionado.

Cuadro 10

Pruebas de variación de la tasa de aprendizaje inicial de DenseNet121

		DenseNet121							
$lr_{inicial}$	**Épocas**	**Luz baja**		**Luz Alta**		**Combinado**		**Promedio**	
		MAE	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**
$10^{-4,4}$	50	60.00	14.93	63.77	13.22	63.86	14.65	62.54	14.27
$10^{-4,3}$	35	55.12	11.25	65.29	11.18	61.82	13.41	60.74	11.95
$10^{-4,2}$	35	55.01	11.59	62.20	14.47	69.10	13.01	62.10	13.03

Nota. La fila sombreada resalta el parámetro seleccionado.

Cuadro 11
Pruebas de variación de la tasa de aprendizaje inicial de Xception.

		Xception							
$lr_{inicial}$	**Épocas**	**Luz baja**		**Luz Alta**		**Combinado**		**Promedio**	
		MAE	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**
$10^{-4,3}$	50	54.47	11.27	66.59	12.77	63.85	12.78	61.64	12.27
$10^{-4,2}$	35	56.86	11.85	61.86	11.76	62.88	12.62	60.53	12.08
$10^{-4,1}$	35	58.63	12.69	62.14	11.54	62.57	12.82	61.12	12.35

Nota. La fila sombreada resalta el parámetro seleccionado.

- ***Número de capas congeladas.*** Se refiere a la cantidad de capas cuyo entrenamiento se desactiva, conservando sus pesos originales.

  Se decide no congelar ninguna capa, siguiendo una estrategia de ajuste fino del transfer learning y basado en los rendimientos mostrados en el Cuadro 12

Cuadro 12
Pruebas de bloques congelados de cada modelo

Nro de bloques congelados	**ResNet50**		**DenseNet121**		**Xception**		**Promedio**	
	MAE	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**	**MAE**	**MAPE**
0	68.97	13.66	61.82	13.41	62.88	12.62	64.56	13.23
1	79.47	18.02	66.28	13.24	70.33	13.88	72.03	15.05
2	126.66	30.95	69.31	14.65	76.15	15.64	90.71	20.41
3	204.53	62.00	108.88	28.50	100.66	19.94	138.02	36.81
4	588.69	100.00	199.46	53.76	201.04	50.19	329.73	67.99

- ***Hiperparámetros del regresor personalizado.***

  - ***Capa de de reducción dimensional.*** Esta capa transforma los datos multidimensionales en un vector 1D, facilitando su entrada en capas densas.

    Se elige la capa Global Average Pooling 2D, basada en el tamaño del archivo generado y el rendimiento mostrado en el Cuadro 13.

Cuadro 13
Pruebas de capas de reducción dimensional

Modelo	**Global Average Pooling**			**Flatten**		
	Tamaño de archivo	**MAE**	**MAPE**	**Tamaño de archivo**	**MAE**	**MAPE**
ResNet50	282MB	68.97	13.66	858MB	79.46	17.73
DenseNet121	87.8MB	61.82	13.41	375MB	73	16.52
Xception	251MB	62.88	12.62	827MB	66	14.11

- ***Número de capas densas.*** Indica cuántas capas completamente conectadas se incluyen en la red, afectando su profundidad y capacidad para aprender características complejas.

  Se elige una sola capa, según los resultados del Cuadro 14, que muestran que no hay mejora sustancial al aumentar el número de capas.

- ***Número de neuronas por capa.*** Especifica la cantidad de neuronas en cada capa densa, determinando la capacidad de aprendizaje y la complejidad del modelo.

  Se seleccionan 512 neuronas, según los resultados del Cuadro 14, que demuestran que no hay mejora significativa al incrementar el número de neuronas.

Cuadro 14

Pruebas de capas y número de neuronas

Capas densas	Numero de neuronas/capa	ResNet50		DenseNet121		Xception		Promedio	
		MAE	MAPE	MAE	MAPE	MAE	MAPE	MAE	MAPE
1	512	68.97	13.66	61.82	13.41	62.88	12.62	64.56	13.23
2	512/512	65.36	13.58	60.19	12.08	64.40	13.07	63.32	12.91
2	1024/512	68.38	14.69	62.51	12.40	64.41	12.78	65.10	13.29
3	1024/512/256	65.81	13.39	55.81	11.61	67.12	14.38	62.91	13.13

- Función de salida. Es la última capa que transforma las activaciones en predicciones finales.

  Se elige la función ReLU, basada en los resultados del Cuadro 15, ya que garantiza predicciones mayores a 0 además de mostrar un buen rendimiento.

Cuadro 15

Pruebas de funciones de salida

Función de salida	ResNet50		DenseNet121		Xception		Promedio	
	MAE	MAPE	MAE	MAPE	MAE	MAPE	MAE	MAPE
Lineal	69.47	13.51	60.82	12.53	64.96	12.74	65.08	12.93
Sigmoide	103.80	26.98	75.00	14.96	72.30	18.25	83.70	20.07
ReLU	68.97	13.66	61.82	13.41	62.88	12.62	64.56	13.23

Detalles Técnicos

Para el desarrollo de este proyecto se hizo uso de los siguientes recursos de hardware y software, los detalles vienen a continuación:

Recursos de Hardware.

Máquina Virtual de Google Colaboratory

- Procesador: Intel(R) Xeon(R) CPU @ 2.00GHz
  - Número de Núcleos: 2 núcleos físicos, 4 hilos de ejecución.
  - Caché: 24.75 MB (L3).
  - Frecuencia Base: 2.00 GHz, con frecuencia turbo de hasta 3.20 GHz.
- RAM: 13 GB
  - Tipo de Memoria: DDR4.
  - Frecuencia: 2133 MHz.
- GPU: NVIDIA Tesla T4
  - Núcleos CUDA: 2560 núcleos CUDA para cálculos paralelos masivos.
  - Memoria VRAM: 16 GB GDDR6 con un ancho de banda de 320 GB/s.
  - Tensor Cores: 320 Tensor Cores.
  - Capacidad de FP16 y INT8.
- Almacenamiento: 80 GB
  - Tipo: SSD (Solid State Drive).

Cámara Fotográfica (Xiaomi Redmi 5 Plus)

- Resolución: 12 megapíxeles.
- Apertura: $f/2{,}2$.
- Tamaño de píxel: 1.25 micrómetros.

- Tamaño del Sensor: 1/2.9 pulgadas
- Distancia Focal Equivalente: 27 mm
- Sensor: CMOS (Complementary Metal-Oxide-Semiconductor).
- Tecnología de Enfoque: Detección de Fase (PDAF-Phase Detection Autofocus).
- Balance de Blancos Automático.
- Control de Exposición: Rango dinámico amplio (HDR).
- Estabilización de Imagen: Electrónica (EIS)
- Vídeo: 4K a 30 FPS y 1080p a 30/60 FPS.
- Rango de ISO: 100 - 3200
- Procesador (System on Chip): Qualcomm Snapdragon 625
    - Número de Núcleos: 8 núcleos (Octa-core).
    - Arquitectura: ARM Cortex-53.
    - Frecuencia de Reloj: 2.0 GHz.
    - Proceso de Fabricación: 14 nm FinFET.
    - GPU Integrada: Adreno 506.
    - Incorpora el ISP (Image Signal Processor) Qualcomm Hexagon DSP, el cual gestiona el procesamiento de imágenes y vídeos.
    - Filtro de Reducción de ruido multi-imagen (MFNR).
    - Capacidad de Codificación y Decodificación de Vídeo:
        - Codificación: H.264, H.265/HEVC, VP8.
        - Decodificación: H.264, H.265/HEVC, VP9, VP8.

Balanza (Ventus B-40)

- Sistema de Pesado: Célula de carga de alta precisión.

- Pantalla Digital: LCD con retroiluminación.
- Función de Tara.
- Estabilización Rápida: Tiempo de estabilización de menos de 3 segundos.

Recursos de Software.

Máquina Virtual de Google Colaboratory

- Sistema Operativo: Ubuntu 18.04.6.
- Lenguaje de programación: Python 3.10.12.
- Librerías: Numpy, Pillow, Matplotlib, Sckit-Image, Tensorflow (2.15.0), Tensorflow-gpu, Keras, Opencv-python.
- Sistema de control de versiones: Git.

Análisis y Discusión de Resultados

En este capítulo final se detallan los resultados de la investigación, revelando los hallazgos derivados de las diversas fases experimentales realizadas para analizar y comparar el rendimiento de los modelos de CNN entrenados en la estimación del peso del cuy. A través de métodos rigurosos y una cuidadosa recopilación de datos, se presenta una visión detallada de las respuestas obtenidas frente a las preguntas de investigación planteadas.

Análisis de Resultados

En esta fase, se presenta el análisis de los resultados del experimento respecto a los datasets de iluminación y los datasets de distorsiones. El análisis se presenta mediante gráficos de entrenamiento, rendimiento y tablas comparativas.

Resultados Basados en los Datasets de Iluminación

En esta sección se evaluaron los efectos de tres conjuntos de datos con diferentes condiciones de iluminación: luz ambiental, luz suplementaria y una combinación de ambas. El objetivo es determinar cuál de estos conjuntos es el más adecuado para entrenar (CNN) en la estimación del peso del cuy, así como analizar el impacto de la iluminación en el rendimiento de los modelos entrenados.

A continuación, se presentan las curvas de entrenamiento individuales de cada uno de los modelos.

Luz Ambiental. En las Figuras 44 y 45, se pueden observar detalles acerca del comportamiento de los modelos durante las épocas de entrenamiento y bajo la influencia de la luz ambiental.

Figura 44

Curvas de entrenamiento de los modelos bajo luz ambiental

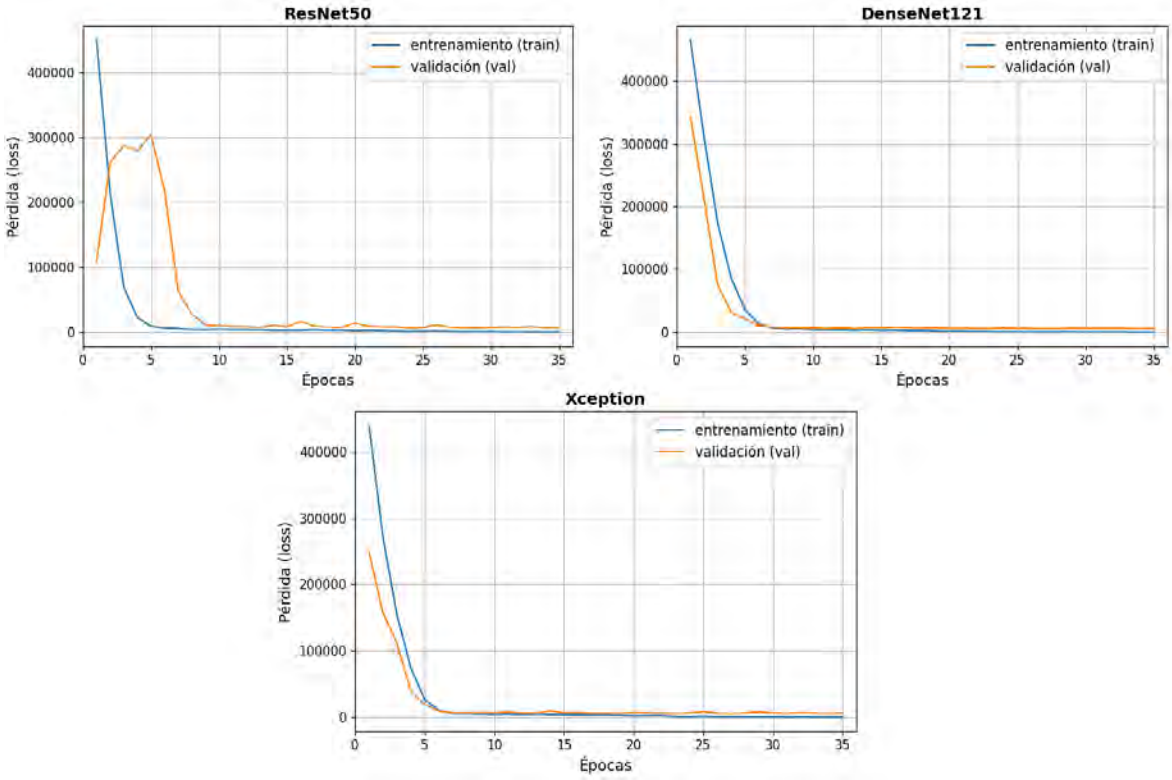
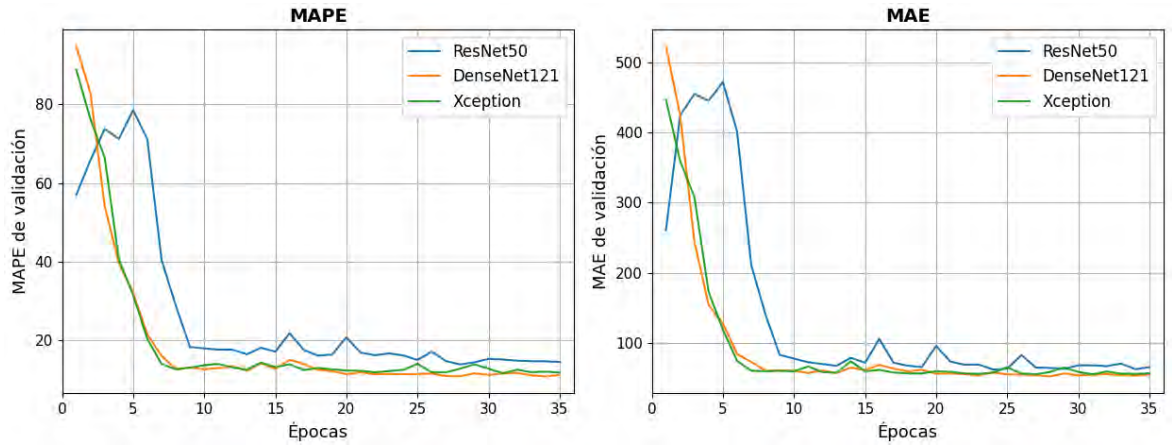


Figura 45

MAPE y MAE de los modelos bajo luz ambiental



- En la Figura 44, se observa que los modelos DenseNet121 y Xception convergen más rápidamente, alcanzando la convergencia antes de la época 10, en comparación con el modelo ResNet, que lo logra mucho después de esta época.
- El modelo ResNet exhibe un error significativamente superior durante las primeras

épocas en comparación con las otras redes, y tarda más en alcanzar los valores mínimos en ambas métricas.

- Es relevante destacar que el modelo ResNet50 muestra picos pronunciados en términos de MAE y MAPE, lo que indica que el modelo enfrenta dificultades durante las épocas tempranas para adaptarse a las imágenes afectadas por la luz ambiental.
- En cuanto al ajuste en el conjunto de validación, en la Figura 45, se destaca que el modelo DenseNet121 logra la mejor adaptación en ambas métricas, aunque con una diferencia mínima en comparación con Xception.

Luz Suplementaria. El comportamiento de los modelos durante las épocas de entrenamiento y bajo la influencia de la luz suplementaria se puede observar en las Figuras 46 y 47.

Figura 46

Curvas de entrenamiento de los modelos bajo luz suplementaria

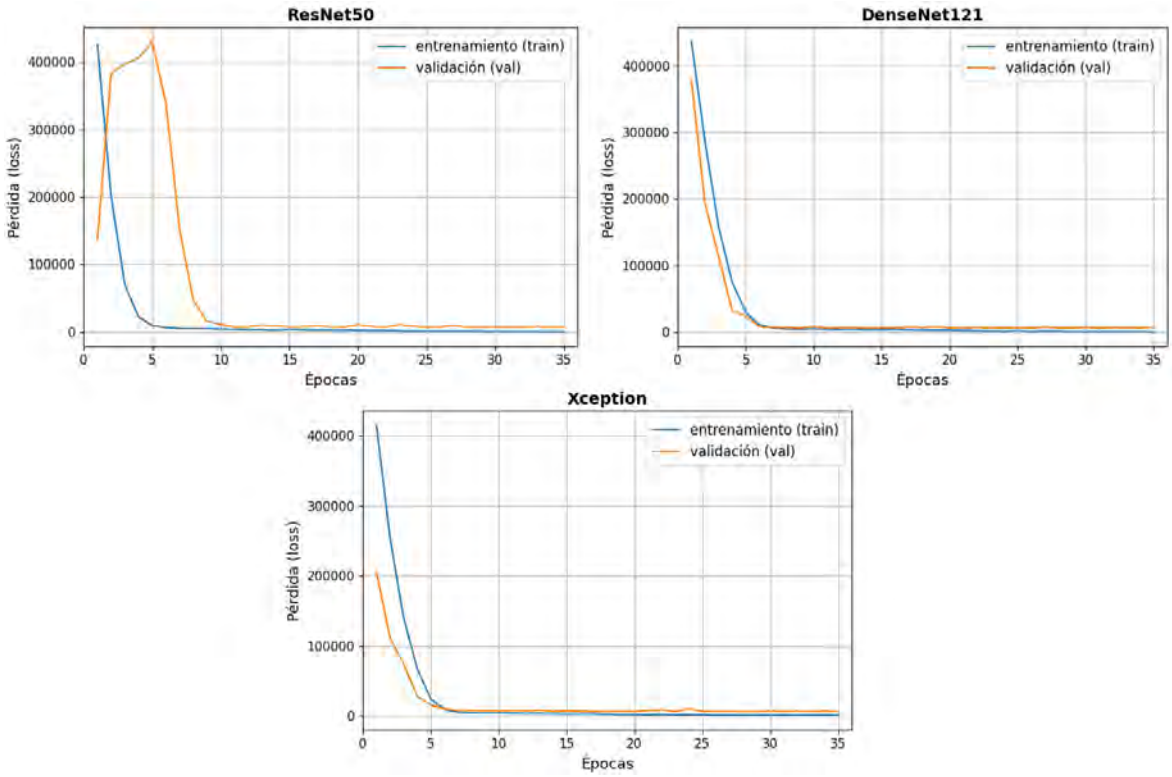
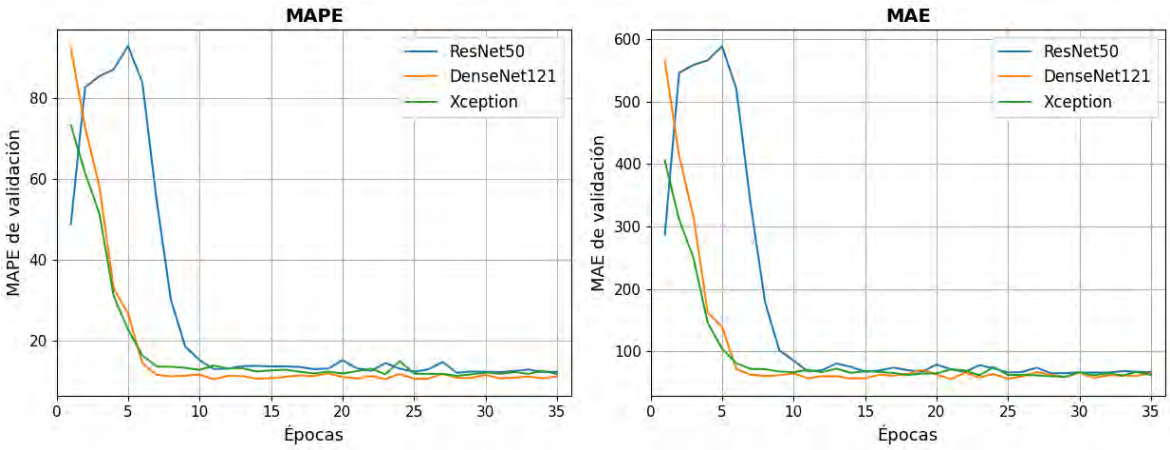


Figura 47

MAPE y MAE de los modelos bajo luz suplementaria



- En la Figura 46, en relación con la convergencia, se evidencia un patrón similar al observado con la luz ambiental, donde DenseNet121 y Xception convergen más rápidamente, antes de la época 10, mientras que ResNet lo hace después.
- En términos de las métricas establecidas, los tres modelos muestran un ajuste muy similar, logrando valores de MAPE por debajo del 20 % en un corto período de épocas y un MAE inferior a 100 gramos durante las primeras 10 épocas.
- En la Figura 47, es relevante destacar que las curvas de los modelos no exhiben fluctuaciones abruptas, indicando que las imágenes con iluminación suplementaria proporcionan un escenario en el cual los modelos se adaptan con mayor facilidad.
- En el conjunto de validación, el modelo DenseNet121 destaca por lograr la mejor adaptación en términos de MAPE, mientras que en relación con el MAE, Xception obtiene el mejor rendimiento.

Prueba Cruzada de Iluminación. Además de las pruebas individuales realizadas para los dos datasets de iluminación, se realiza una prueba cruzada para evaluar el rendimiento de los modelos con un dataset distinto al de sus datos de entrenamiento y validación. Este experimento es crucial, ya que los resultados obtenidos resaltan la importancia del estudio del dataset combinado. A continuación, en el Cuadro 16, se presentan los resultados de esta prueba:

Cuadro 16

Resultados de la prueba cruzada por tipo de iluminación.

Modelo	Dataset de entrenamiento	Dataset de prueba			
		Luz ambiental		Luz suplementaria	
		MAE	MAPE	MAE	MAPE
ResNet50	Luz ambiental	80.47	16.15	76.94	15.61
DenseNet121		65.45	12.06	80.25	16.96
Xception		83.70	17.42	77.09	17.06
ResNet50	Luz suplementaria	189.72	37.93	67.04	12.92
DenseNet121		161.21	35.61	61.14	10.80
Xception		149.82	32.50	60.28	11.92

Nota. Las celdas sombreadas resaltan los resultados de la prueba cruzada contra las pruebas originales.

De los resultados a partir de las pruebas cruzadas, se pueden extraer las siguientes interpretaciones:

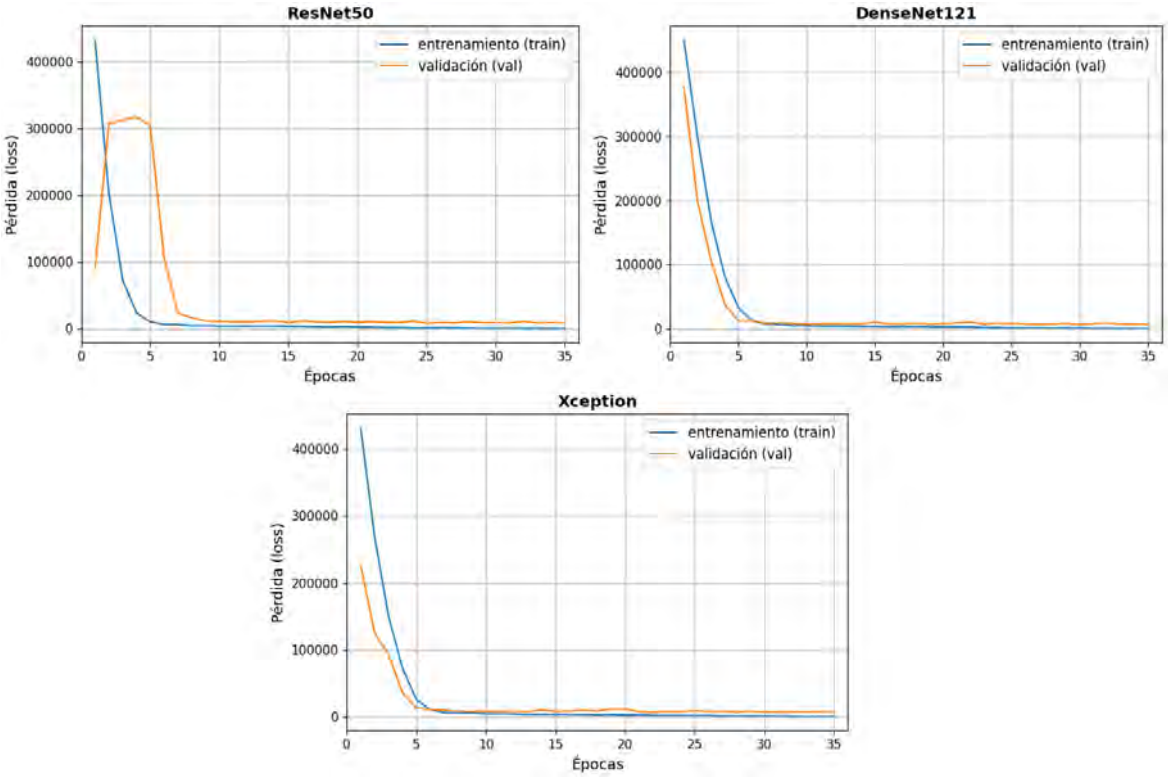
- Los modelos entrenados con luz ambiental demuestran una notable capacidad de adaptación a la luz suplementaria. Esto se evidencia al observar que, en términos de MAPE, los modelos ResNet50 y Xception mejoran en un 0.54 % y 0.36 %, respectivamente, mientras que DenseNet121 solo incrementa su error en 15.20 gramos en términos de MAE. Este fenómeno indica que los modelos entrenados con luz ambiental logran aprender a extraer mejores patrones e información más significativa de este tipo de imágenes, permitiéndoles predecir el peso del cuy incluso en presencia de sombras y una iluminación menos intensa.
- En contraste, los modelos entrenados con luz suplementaria presentan dificultades significativas para adaptarse a la luz ambiental. Los aumentos en los errores MAE y MAPE son notables, con incrementos de hasta 122 gramos y 25.01 % para ResNet50, 100.08 gramos y 24.81 % para DenseNet121, y 89.54 gramos y 20.58 % para Xception. Este patrón indica que bajo la luz suplementaria, el proceso de ajuste es demasiado simplificado y los modelos se adaptan excesivamente a los datos de entrenamiento, lo que dificulta generalizar las características esenciales para la estimación precisa del peso del cuy en situaciones diversas.

- Como consecuencia de lo observado anteriormente, surge la pregunta de si un modelo entrenado con la combinación de imágenes con ambos tipos de iluminación podría ofrecer un rendimiento mejorado en la predicción del peso. Esta cuestión se explora en las secciones posteriores.

Combinado. En las Figuras 48 y 49, se proporcionan detalles adicionales sobre el rendimiento de los modelos durante su entrenamiento con el conjunto de datos combinado.

Figura 48

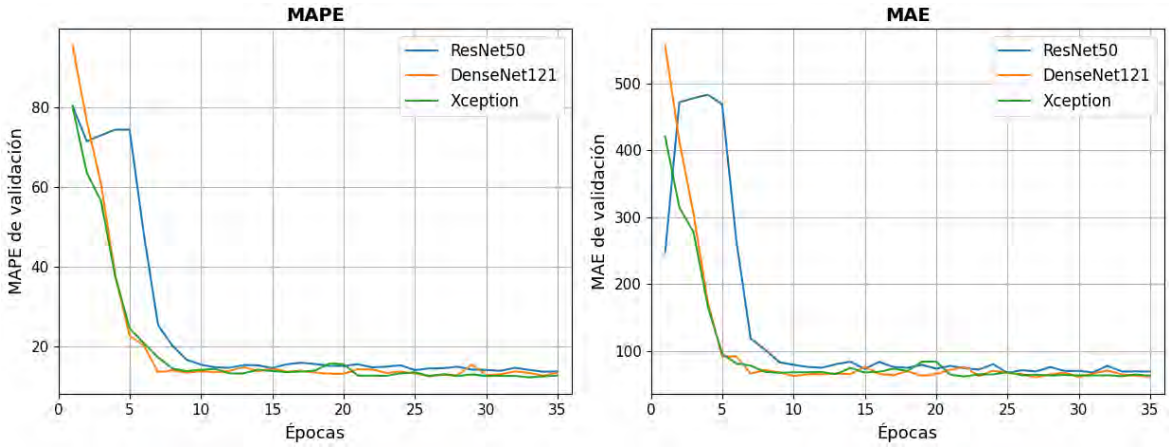
Curvas de entrenamiento de los modelos bajo el dataset combinado



- En la Figura 48, se destaca que el modelo ResNet50 logra la convergencia antes de la época 10, lo cual representa una mejora significativa en el tiempo de convergencia para este modelo en comparación con los dos conjuntos de datos anteriores.
- En la Figura 49, que presenta las métricas, se observa que las fluctuaciones en las curvas se suavizan en cierta medida. Esto sugiere que al entrenar los modelos con una combinación de imágenes con distintos tipos de iluminación, se genera un error menos variable, resultando en modelos más versátiles frente a una amplia variedad de situaciones.

Figura 49

MAPE y MAE de los modelos bajo el dataset combinado



- En el conjunto de validación, se destaca que el modelo Xception muestra la mejor adaptación en términos de MAPE, mientras que en relación con el MAE, DenseNet121 logra el mejor rendimiento.

Resultados Finales. Los resultados finales del entrenamiento de los modelos bajo los tres datasets de iluminación se presentan en el Cuadro 17:

Cuadro 17

Resultados finales del entrenamiento de los tres modelos para los datasets de iluminación.

Dataset	Modelo	Validación		Prueba	
		MAE	**MAPE**	**MAE**	**MAPE**
Luz ambiental	ResNet50	65.59	14.51	80.47	16.15
	DenseNet121	55.12	11.25	65.45	12.06
	Xception	59.35	12.47	83.70	17.42
Luz suplementaria	ResNet50	66.54	12.31	67.04	12.92
	DenseNet121	65.29	11.18	61.14	10.80
	Xception	61.86	11.76	60.28	11.92
Combinado	ResNet50	68.97	13.66	73.49	14.43
	DenseNet121	61.82	13.41	64.15	13.18
	Xception	62.88	12.62	73.73	15.20

Nota. Las celdas sombreadas representan los valores óptimos para cada métrica para cada dataset

Algunos de los principales hallazgos que se pueden derivar del Cuadro 17 son los siguientes:

Sobre la iluminación:

- En cuanto a los dos tipos de iluminación, es destacable que, según los resultados obtenidos por los tres modelos, la luz suplementaria es un escenario más propicio para el ajuste que la luz ambiental. Esto se debe a la ausencia significativa de sombras y variaciones abruptas de iluminación en las imágenes con este tipo de luz. La diferencia en el MAPE de los modelos al comparar la iluminación ambiental con la suplementaria es evidente: ResNet50 registra 16.15 % y 12.92 %, DenseNet121 12.06 % y 10.80 %, y Xception 17.42 % y 11.92 %, respectivamente.
- En relación con la luz ambiental, es importante resaltar que representa un escenario bastante complejo para los modelos. Aunque este entorno es complicado, mejora la capacidad de los modelos para captar las características esenciales de las imágenes de los cuyes, lo que contribuye a una predicción más precisa del peso del animal. Esto se demuestra en las pruebas cruzadas presentadas en el Cuadro 16.
- Basándonos en los resultados de la prueba cruzada en el Cuadro 16 y en los resultados finales en el Cuadro 17, podríamos afirmar, con mínimas diferencias, que los modelos entrenados con el conjunto de datos combinado ofrecen el mejor rendimiento para la predicción del peso del cuy.

Sobre los modelos:

- DenseNet121 destaca como el modelo con los mejores resultados en los tres conjuntos de datos, ya que presenta consistentemente el menor error promedio grupal (MAE) y el menor error en proporción al peso original de cada cuy (MAPE).
- Aunque ResNet50 y Xception, individualmente, no generalizan bien con luz ambiental, su buen desempeño con luz suplementaria los acerca en cierta medida a un rendimiento aceptable cuando se evalúan con el conjunto de datos combinado.
- En el caso de la luz suplementaria, destacan dos modelos por su rendimiento. La

arquitectura Xception muestra un error promedio de alrededor de 60 gramos, indicando un ajuste notable en la minimización del error. Por otro lado, DenseNet121 exhibe un destacado 10 % en términos de la proporción entre el error de cada cuy y su peso original, señalando una mejor adaptación para predecir pesos y evitar errores significativos.

- Respecto a la combinación de imágenes con diferentes niveles de iluminación en el conjunto de datos combinado, el modelo más acertado resulta ser DenseNet121. Este modelo, como era de esperar, muestra un rendimiento satisfactorio en los otros conjuntos de datos por separado. Además, es el único modelo que presenta variaciones mínimas en el error absoluto promedio en comparación con sus competidores, con valores de 65.45 gramos, 61.14 gramos y 64.15 gramos para luz ambiental, luz suplementaria y el conjunto de datos combinado, respectivamente.

Resultados Basados en el Dataset Principal

Tras las pruebas de iluminación y los resultados obtenidos, se generó el dataset principal. Este conjunto ampliado incorpora imágenes adicionales de cuyes y nuevos pesos, lo que aumenta la complejidad del desafío de predecir el peso de los cuyes mediante los modelos.

Los ajustes se efectúan en el hiperparámetro de la "tasa de aprendizaje" (ver Ecuación 16) para cada modelo. Las modificaciones en su cálculo son las siguientes:

- Las etapas de decaimiento (ds) se establecen en 3000 para los tres modelos, considerando la triplicación en la cantidad de imágenes.
- La tasa de aprendizaje inicial ($lr_{inicial}$) se incrementa proporcionalmente a partir de los valores óptimos obtenidos anteriormente. Por lo tanto, el valor final para ResNet50 se ajusta a 10^{-5}, para DenseNet121 a $10^{-4,6}$ y para Xception a $10^{-4,5}$.

  Los resultados del entrenamiento se presentan a continuación:

Figura 50

Curvas de entrenamiento de los modelos bajo el dataset principal

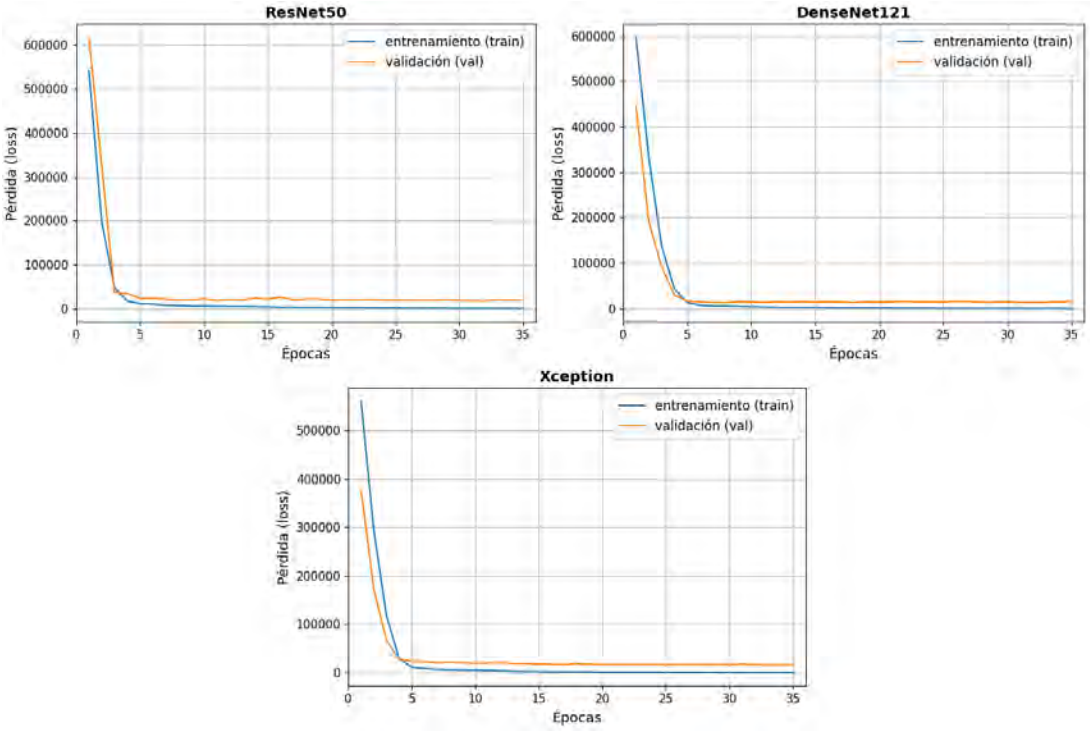
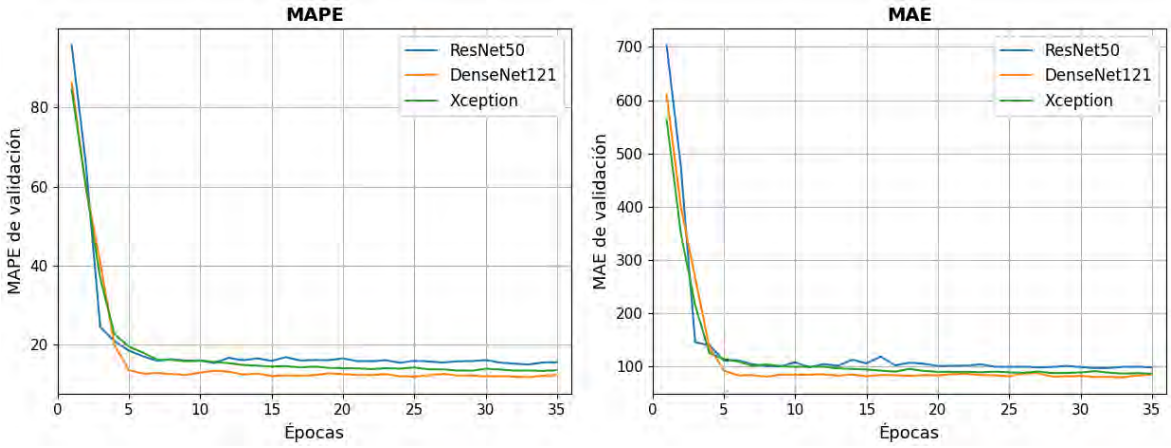


Figura 51

MAPE y MAE de los modelos bajo el dataset principal



En cuanto al proceso de entrenamiento y las métricas de los modelos, al observar las Figuras 50 y 51, se destacan los siguientes puntos:

- Todos los modelos logran la convergencia antes de alcanzar la época 5, incluido el modelo ResNet50, que previamente convergía cerca de la época 10 con el conjunto de datos combinado. Este cambio se atribuye al reciente ajuste de la tasa de

aprendizaje y al aumento en la cantidad de datos disponibles.

- En términos de la métrica MAPE, la diferencia en los errores entre los tres modelos se hace más evidente. DenseNet121 alcanza el menor error de manera más rápida y mantiene un descenso constante hasta el final. Le sigue de cerca Xception, mientras que ResNet50 muestra el mayor error al final del entrenamiento debido a sus dificultades para un ajuste más fino.

- ResNet50, además de tener el mayor error en términos de MAPE y MAE, exhibe desajustes abruptos a lo largo de las épocas, especialmente notables en las curvas MAE. Esto sugiere que el modelo enfrenta dificultades para realizar un ajuste preciso. Esta misma observación se había registrado previamente en el conjunto de datos de luz ambiental, lo que indica que este comportamiento puede deberse al aumento de imágenes con esas condiciones en el conjunto de datos actual.

En el Cuadro 18 se ofrece una evaluación final del desempeño de los modelos contrastando los resultados del dataset combinado y del dataset principal.

Cuadro 18

Comparación del dataset combinado con el dataset principal

Dataset	Modelo	Validación		Prueba	
		MAE	**MAPE**	**MAE**	**MAPE**
Combinado	ResNet50	68.97	13.66	73.49	14.43
	DenseNet121	61.82	13.41	64.15	13.18
	Xception	62.88	12.62	73.73	15.20
Principal	ResNet50	98.49	15.58	104.68	16.23
	DenseNet121	84.77	12.36	87.10	12.84
	Xception	86.77	13.63	95.01	14.78

- Al revisar el Cuadro 18, se observa que los modelos DenseNet121 y Xception muestran mejoras en sus errores MAPE gracias al incremento en la cantidad de imágenes y pesos del conjunto de datos principal. Sus errores MAPE disminuyen de 13.18 % a 12.84 % y de 15.20 % a 14.78 %, respectivamente.

- En cambio, el modelo ResNet50 experimenta un aumento en sus errores MAE y MAPE con el conjunto de datos principal. Este incremento puede atribuirse a la inclusión de nuevas imágenes que presentan sombras y una iluminación irregular,

fenómeno similar al observado con la luz ambiental. Como se ha señalado anteriormente, estas condiciones lumínicas tienen un impacto considerable en el desempeño de los tres modelos. A partir de estas observaciones, podemos concluir que ResNet50 se ve afectado de manera más pronunciada por la luz ambiental en comparación con sus contrapartes.

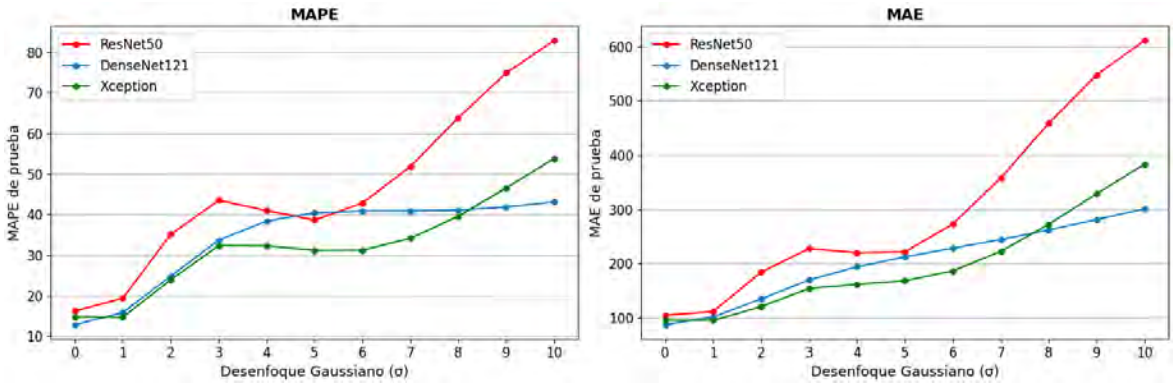
Resultados Basados en los Datasets de Distorsiones

En esta sección, inicialmente se evalúa el rendimiento de cada modelo a través de todos los niveles de distorsiones, empleando las métricas MAE y MAPE. A continuación, se analizan algunas imágenes con cada modelo mediante la técnica de visualización GradCAM (Selvaraju *et al.*, 2019). GradCAM facilita la comprensión de las áreas clave en una imagen que influyen en las decisiones de un modelo de red neuronal convolucional, generando un mapa de calor que resalta las regiones más relevantes al combinar las activaciones ponderadas de la red. Esta representación visual es útil para interpretar los modelos y mejorar la confianza en los resultados obtenidos.

Resultados Basados en el Desenfoque Gaussiano. Los resultados del rendimiento general de los modelos se presentan en la Figura 52 a continuación:

Figura 52

Curvas de rendimiento de los modelos bajo distintos niveles de desenfoque gaussiano



Nota. El nivel 0 indica ausencia de la distorsión.

Sobre la distorsión:

- El desenfoque gaussiano muestra un impacto moderado en los modelos de CNN. Tanto DenseNet121 como Xception exhiben las mejores tolerancias durante los niveles bajos (primer tercio de niveles). Dada la naturaleza de la tarea, estos aumentos en el error se consideran moderados hasta alcanzar un nivel aceptable de distorsión para un observador humano (entre $\sigma = 5$ y $\sigma = 7$), donde provocan incrementos no mayores al 50 % con respecto al peso original de cada cuy. Al llevar la distorsión al máximo nivel, estos incrementos de error ya superan la barrera del 50 % de MAPE.

Sobre los modelos:

- En las gráficas de MAPE y MAE de la Figura 52, se evidencia una tendencia en la que DenseNet121 y Xception exhiben un rendimiento similar durante el primer tercio de niveles. En el segundo tercio, en términos de MAPE, Xception mantiene un nivel casi constante de alrededor del 30 % y DenseNet en 40 %, mientras que en el último tercio DenseNet logra superar a Xception manteniéndose por debajo del 45 % y 300 gramos de MAPE y MAE, respectivamente.
- En el caso de ResNet50, este modelo se ve significativamente afectado incluso por niveles bajos de desenfoque gaussiano. Al probarlo con imágenes que tienen el nivel más alto de desenfoque, se registra un error superior al 80 % o 600 gramos en promedio.

A continuación, se realiza una evaluación visual de algunos ejemplos de esta distorsión:

Figura 53

Primer ejemplo del análisis del desenfoque gaussiano.

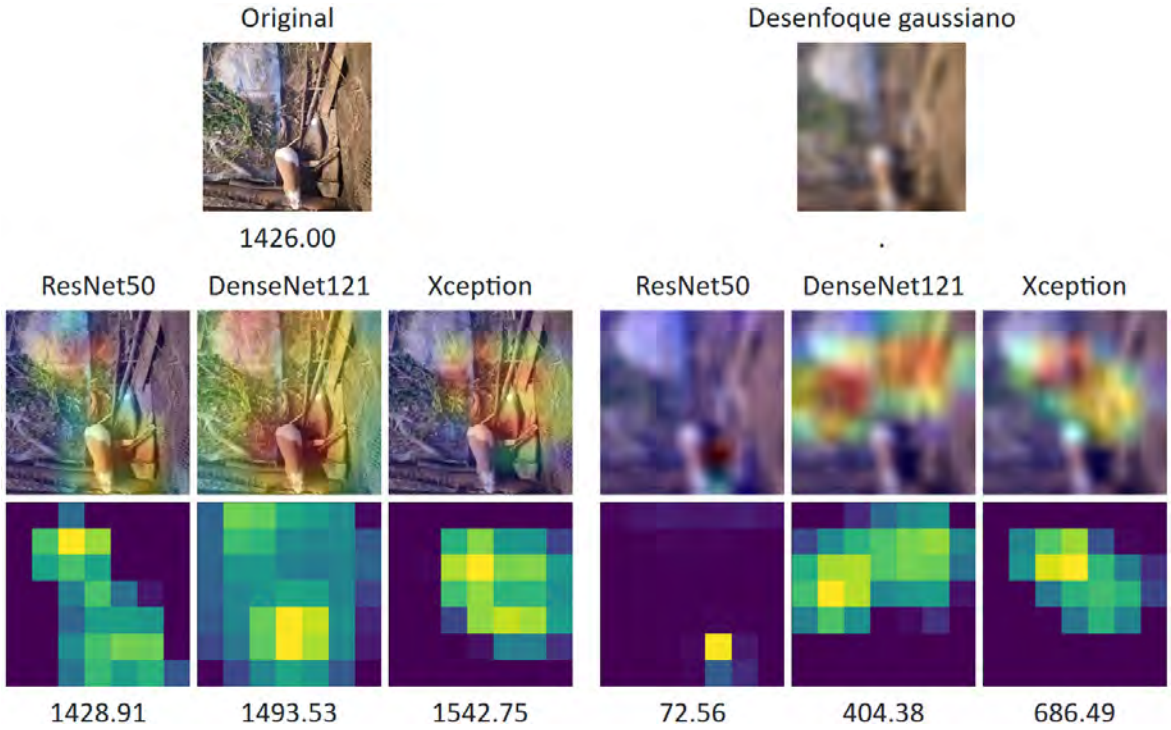
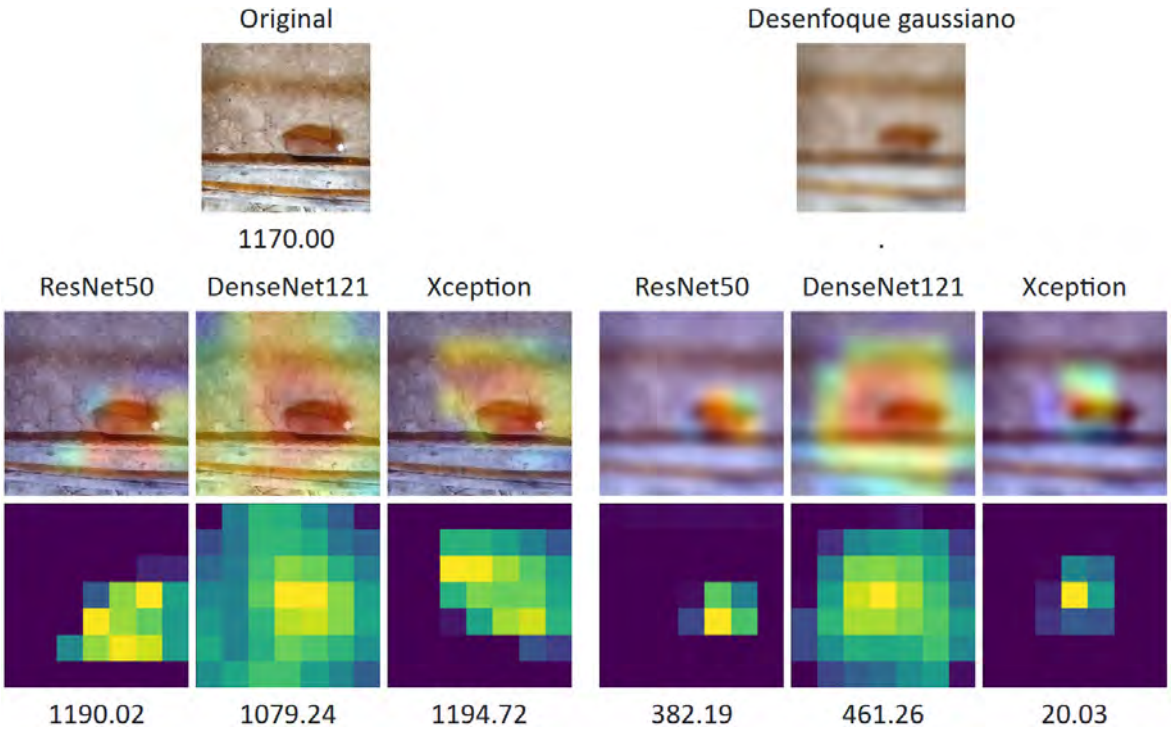


Figura 54

Segundo ejemplo del análisis del desenfoque gaussiano.



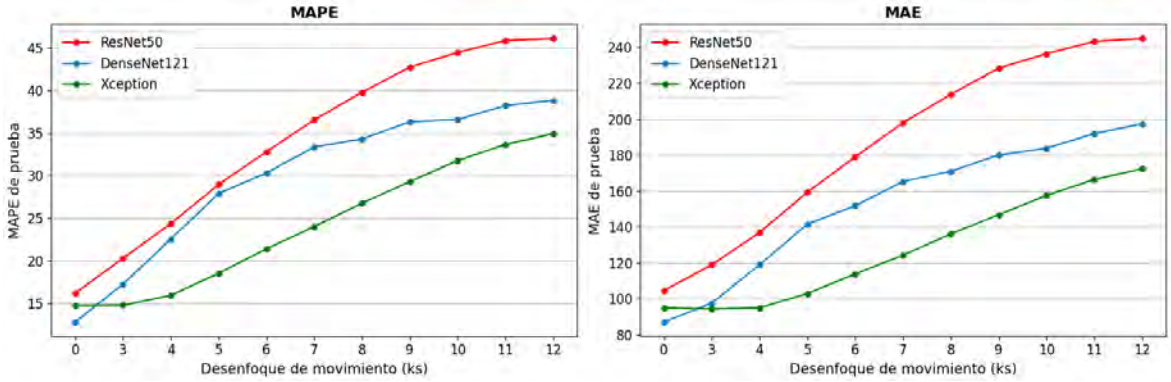
De las Figuras 53 y 54, se pueden derivar las siguientes interpretaciones:

- El desenfoque gaussiano provoca la pérdida total de las texturas en las imágenes. No obstante, en ciertos ejemplos, como el de la Figura 54, donde el entorno que rodea al cuy no es muy complejo, las redes, como ResNet50 y DenseNet121, logran identificar la posición del animal. Sin embargo, la afectación en la textura del cuy repercute en predicciones menos precisas.
- En el caso específico de la Figura 53, se aprecia que la predicción de las redes se ve completamente comprometida, ya que no logran identificar al cuy en la imagen y recurren a otras áreas en búsqueda de texturas similares a las aprendidas durante el entrenamiento para realizar la predicción.
- En resumen, en relación al desenfoque gaussiano, se puede destacar que esta distorsión afecta principalmente a las texturas de las imágenes, las cuales son esenciales para identificar al cuy y realizar predicciones precisas.

Resultados Basados en el Desenfoque de Movimiento. Los resultados del rendimiento general de los modelos se presentan en la Figura 55 a continuación:

Figura 55

Curvas de rendimiento de los modelos bajo distintos niveles de desenfoque de movimiento



Nota. El nivel 0 indica la ausencia de la distorsión.

Sobre la distorsión:

- En general, el desenfoque de movimiento afecta de manera moderada a las redes, destacando especialmente la resistencia de la red Xception. En el nivel máximo de distorsión, Xception muestra un error de solo 35 % en MAPE y 180 gramos en

MAE, mientras que las otras dos redes presentan errores superiores al 47 % y los 245 gramos, respectivamente.

Sobre los modelos:

- Al comparar el rendimiento de los modelos a lo largo de todos los niveles, se destaca que Xception es, con diferencia, el modelo más resistente al desenfoque de movimiento, seguido por DenseNet121 y, en última instancia, ResNet50.
- Analizando las curvas MAE y MAPE, se interpreta que ResNet50 y DenseNet121 no son tan resilientes desde los niveles más bajos ($3 \leq ks < 7$) del desenfoque de movimiento, por lo que no se consideran las mejores opciones al trabajar con imágenes que contengan esta distorsión.
- Xception se ha mostrado como el modelo más adecuado para enfrentar el desenfoque de movimiento. En los dos niveles más bajos de distorsión (ks=3 y ks=4), solo presenta un aumento del 1 % en el error de predicción. En contraste, para el nivel máximo, el aumento en el error es del 20 %, comparado con el 26 % en DenseNet121 y casi el 30 % en ResNet50.

Figura 56

Primer ejemplo del análisis visual del desenfoque de movimiento.

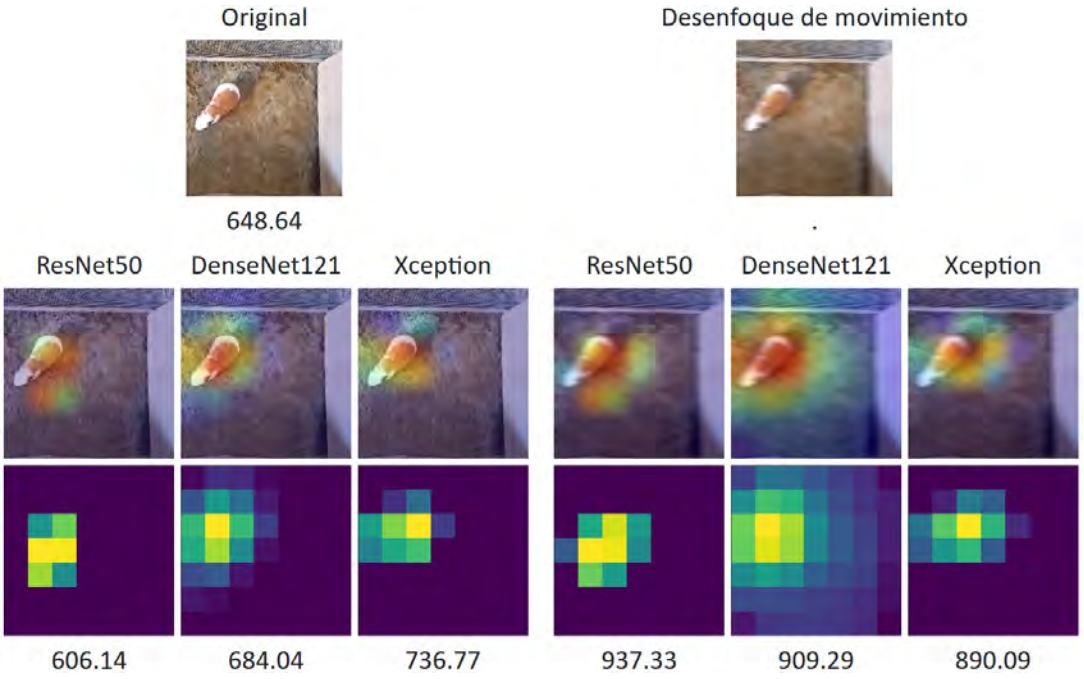
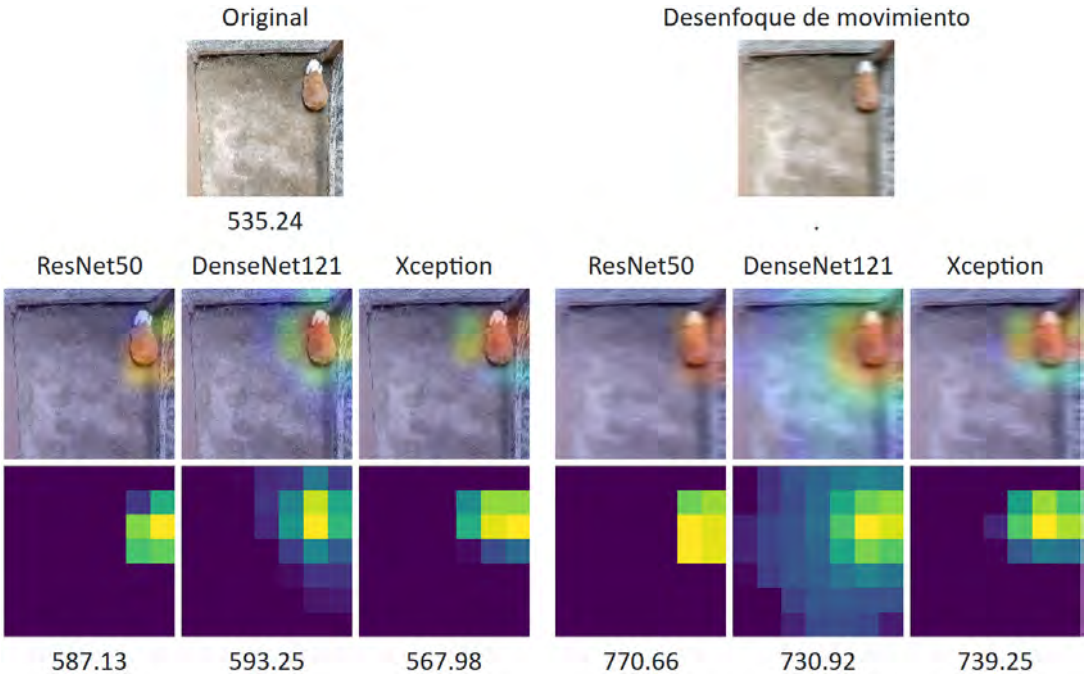


Figura 57

Segundo ejemplo del análisis visual del desenfoque de movimiento.



Algunas observaciones de las Figuras 56 y 57 son:

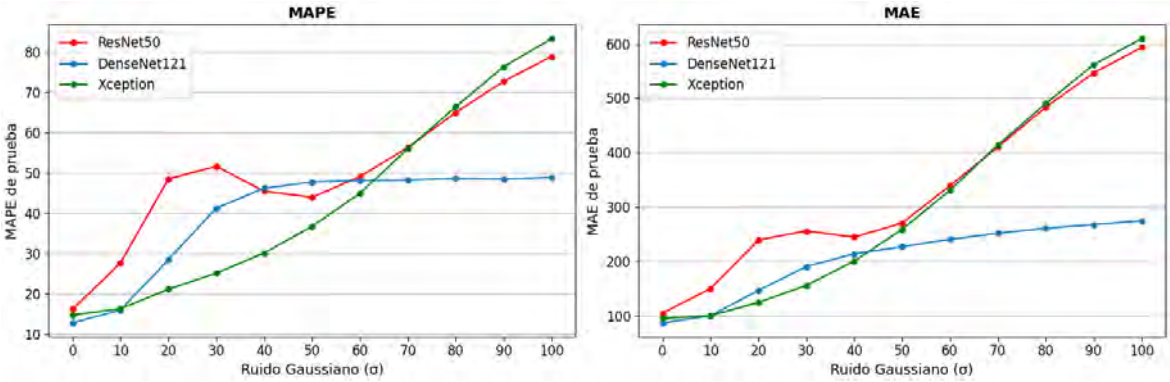
- En ambas figuras, se observa que el desenfoque de movimiento provoca que el área ocupada por el cuy se vea incrementada debido al arrastre de píxeles causado por esta distorsión.
- Comparando los mapas de calor de las imágenes distorsionadas con los de las imágenes originales, se nota la adición de características nuevas tenidas en cuenta por todos los modelos. Respecto a esto, DenseNet121 se ve afectado en gran medida, ya que esta red, debido a su arquitectura interconectada, considera una mayor cantidad de características para predecir el peso.
- En el caso de Xception, se puede observar que esta red no se ve altamente afectada por esta distorsión, ya que toma en consideración características del cuy que no se ven afectadas altamente por el desenfoque de movimiento.
- En la Figura 57, al observar los mapas de calor de ResNet50, se aprecia que esta red se ve afectada porque las características que considera esenciales se multiplican, lo que provoca un aumento en el peso predicho.

- En resumen, se puede determinar que el desenfoque de movimiento afecta directamente al área que ocupa el animal, causando que en la mayoría de los casos los modelos sobreestimen el peso del cuy.

Resultados Basados en el Ruido Gaussiano. Los resultados del rendimiento general de los modelos se presentan en los gráficos a continuación:

Figura 58

Curvas de rendimiento de los modelos bajo distintos niveles de ruido gaussiano



Nota. El nivel 0 indica que no hay presencia de la distorsión.

Sobre la distorsión:

- La presencia de ruido gaussiano afecta significativamente el rendimiento de los modelos en la estimación del peso de los cuyes. Este tipo de ruido puede hacer que los modelos aumenten las predicciones de peso en más del 20 % en promedio, incluso con niveles bajos de ruido ($\sigma = 20$). En niveles extremos de ruido, ResNet50 y Xception superan el 75 % de error promedio, resultando en un aumento de casi 600 gramos en el peso estimado.

Sobre los modelos:

- En términos generales, al examinar las curvas de MAPE y MAE, se observa que el modelo más resistente a la influencia del ruido gaussiano resulta ser DenseNet121, manteniendo su error por debajo del 50 % y los 300 gramos en promedio, incluso en los niveles más altos de ruido.
- En cuanto a MAPE, hasta una desviación estándar $\sigma = 60$, Xception supera

notablemente a los otros dos modelos, indicando que es más adecuado cuando las imágenes presentan una presencia mínima de ruido. Sin embargo, al superar el nivel $\sigma = 70$, su rendimiento se vuelve incluso peor que el de ResNet50.

- ResNet50 es el modelo más afectado por el ruido gaussiano en general. Aunque se observa un ligero descenso en el error en los niveles $\sigma = 40$ y $\sigma = 60$ en el gráfico de MAPE, este descenso podría ser un artefacto de la aleatoriedad en la generación del ruido, ya que el error aumenta nuevamente y se iguala al de Xception. Esta tendencia también se observa en el gráfico de MAE, donde la disminución del error solo ocurre en $\sigma = 40$, volviendo luego a la tendencia general.

Figura 59

Primer ejemplo del análisis del ruido gaussiano.

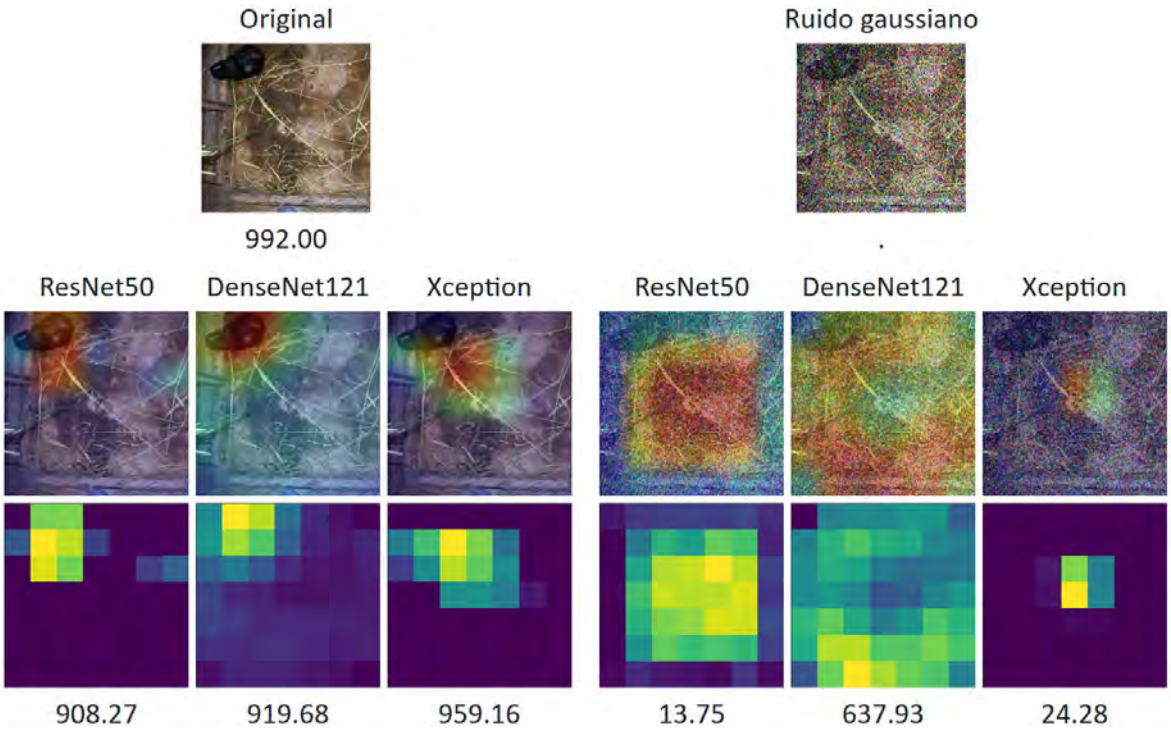
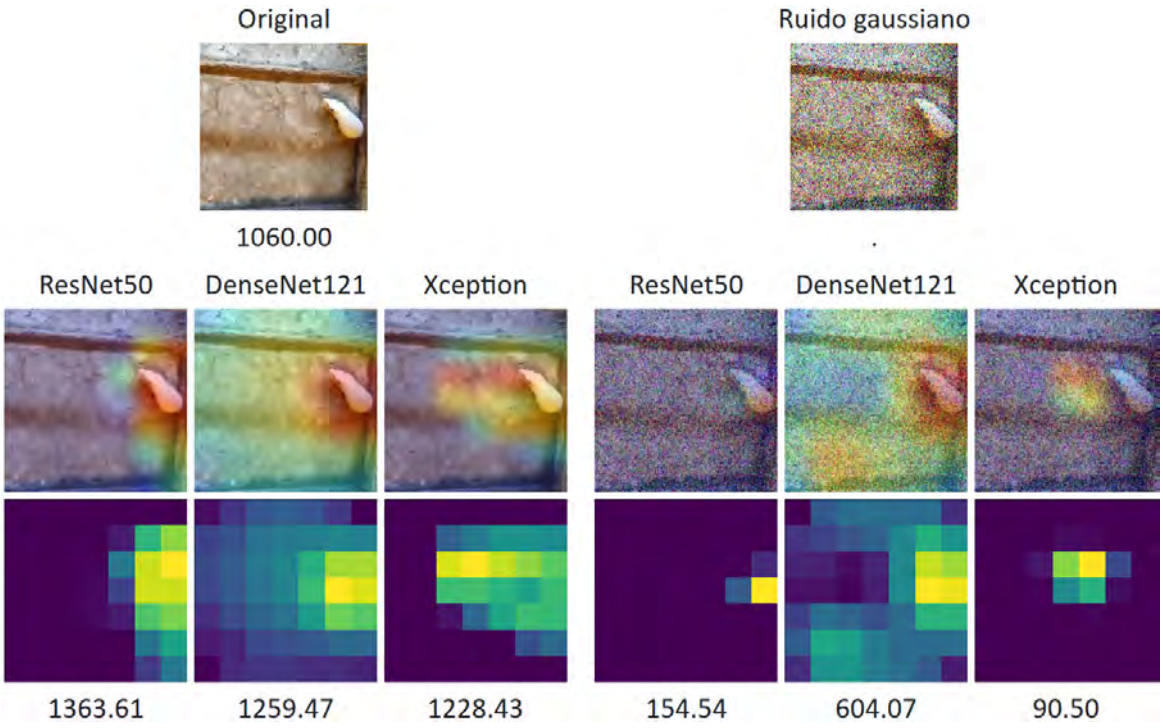


Figura 60

Segundo ejemplo del análisis del ruido gaussiano.



En las Figuras 59 y 60, donde se contrastan las imágenes originales (izquierda) con las mismas imágenes pero con el nivel máximo de ruido gaussiano ($\sigma = 100$) (derecha), se destacan los siguientes aspectos:

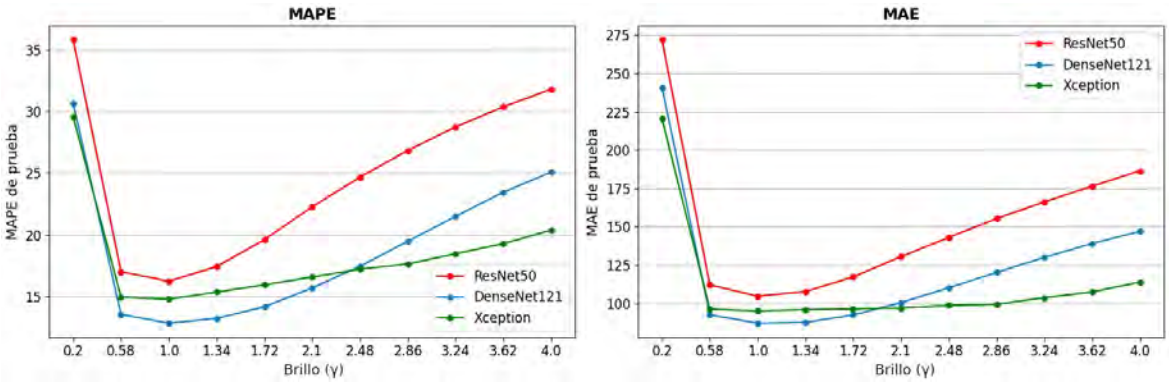
- En términos generales, los modelos exhiben una notable sensibilidad al ruido, ya que esta distorsión tiene un efecto de confusión sobre las redes, dificultándoles la identificación clara del contorno del ejemplar de cuy en la imagen.
- Al examinar las evaluación de las imágenes distorsionadas de ambos ejemplos, ResNet50 y Xception muestran muy poca resiliencia al ruido, ya que no logran identificar las características esenciales del cuy, resultando en predicciones muy bajas, incluso por debajo de 150 gramos.
- DenseNet121, al considerar una mayor cantidad de regiones de la imagen, logra identificar mejores características del cuy, generando mejores predicciones en comparación con sus contrapartes.
- En resumen, se puede concluir que el ruido gaussiano impide que las redes

neuronales convolucionales identifiquen los contornos de los objetos, lo que resulta en la pérdida de la precisión necesaria para estimar adecuadamente el peso animal.

Resultados Basados en el Brillo. Los resultados del rendimiento general de los modelos frente a distintas variaciones de brillo se muestran en la Figura 61.

Figura 61

Curvas de rendimiento de los modelos bajo distintos niveles de brillo



Nota. El nivel 1 indica ausencia de la distorsión.

Sobre la distorsión:

- Las variaciones de brillo se manifiestan de dos maneras: cuando el factor γ varía entre los valores 0 y 1, lo que oscurece la imagen (reducción de brillo), y cuando el factor γ varía entre los valores 1 y 4, lo que aclara la imagen (incremento de brillo).
- En términos generales, las variaciones de brillo no generan un impacto muy significativo en los modelos, ya que, en comparación con otras distorsiones, el error promedio de predicción apenas supera el 35 % y los 275 gramos durante los niveles de reducción de brillo.
- El brillo afecta más a las arquitecturas en los niveles de oscurecimiento que en los niveles de aclaramiento. Esta diferencia en el impacto se refleja en un aumento de aproximadamente 100 gramos en el error de predicción.

Sobre los modelos:

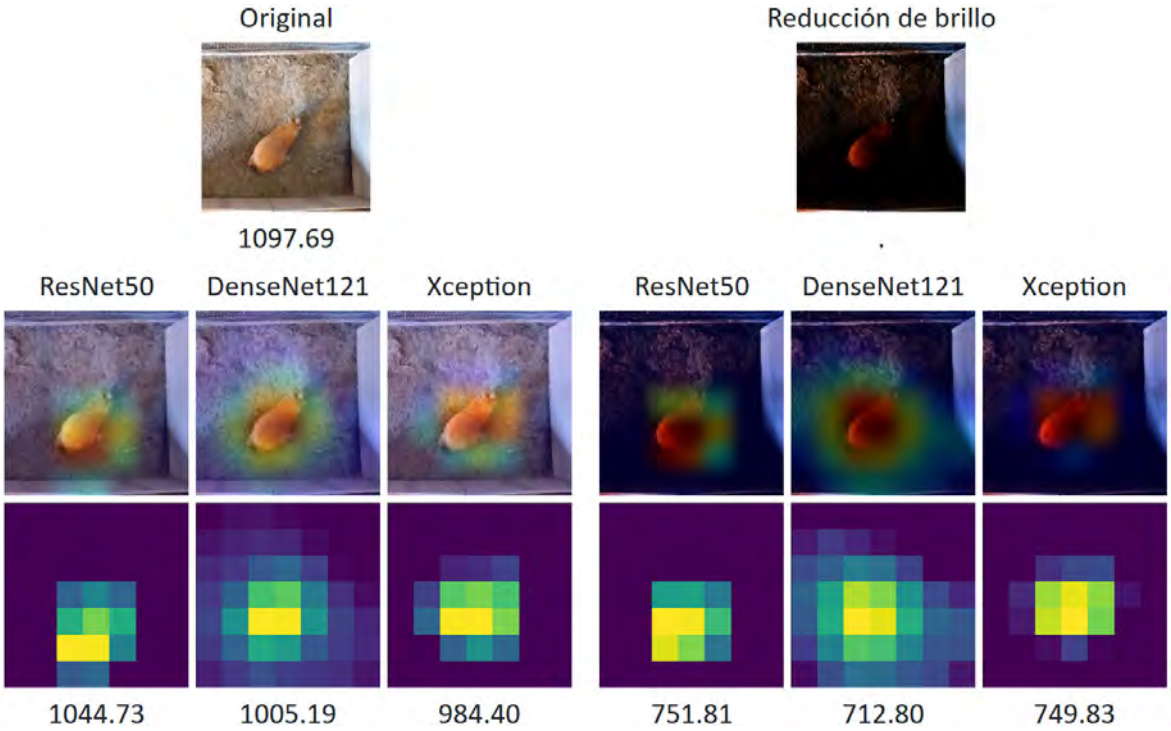
- En general, considerando el MAPE, DenseNet121 y Xception muestran una baja sensibilidad a los niveles bajos de reducción e incremento de brillo

$(0{,}58 \leq \gamma \leq 2{,}48)$, con un aumento en el error de menos del 5 % en estos niveles. DenseNet121 demuestra una mayor resiliencia en comparación con Xception en estos niveles bajos.

- Al considerar los niveles altos de reducción e incremento de brillo ($0 < \gamma < 0{,}58$ y $2{,}48 < \gamma$), DenseNet121 y Xception ya se ven más afectados por estas variaciones, siendo Xception el que mantiene el menor error en términos de MAE y MAPE.
- ResNet50 se ve considerablemente afectado en cualquier nivel de incremento o reducción de brillo. Aunque en las fases de oscurecimiento iniciales ($0{,}58 \leq \gamma < 1$) muestra una cierta resiliencia.

Algunas evaluaciones visuales se presentan en las Figuras 62 y 63 a continuación:

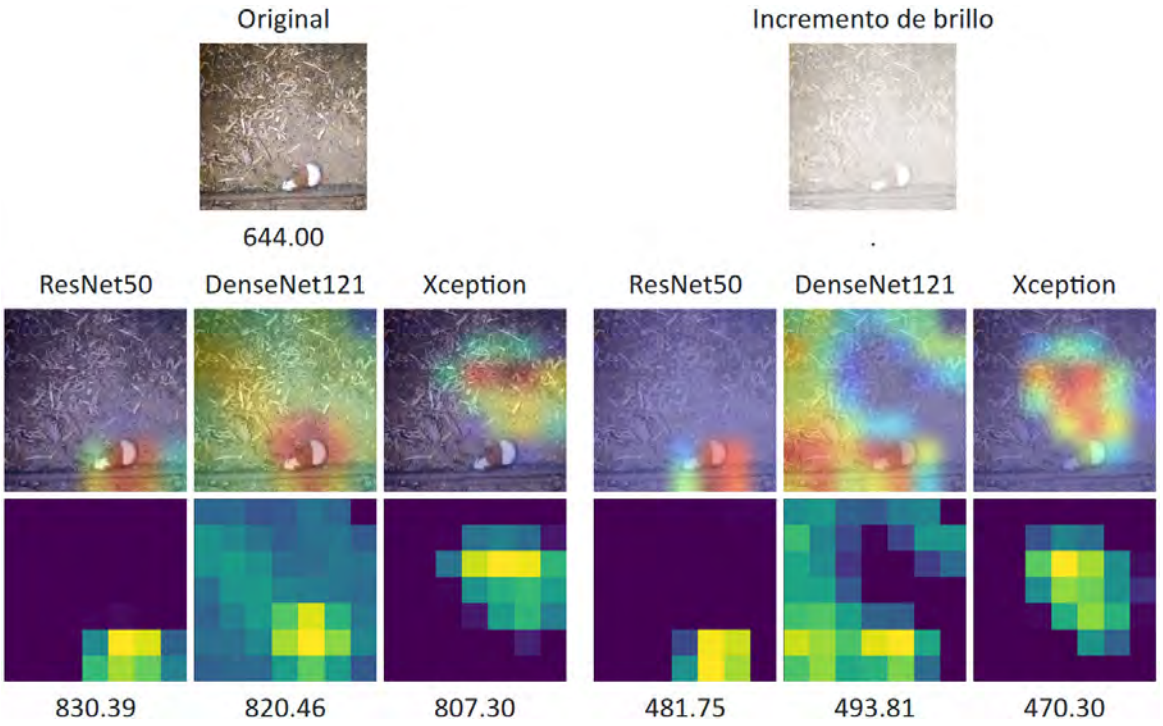
Figura 62
Ejemplo del análisis de la reducción de brillo.



- En la Figura 62, se presenta el ejemplo de una imagen con el nivel máximo de reducción de brillo considerado ($\gamma = 0{,}2$). Para este caso, se observa que las tres redes aún logran identificar al animal, pero sus colores se ven oscurecidos y se confunden con el fondo. Esto resulta en una menor cantidad de características

Figura 63

Ejemplo del análisis del incremento de brillo.



consideradas por los modelos. Tal como se observa en los mapas de calor y las imágenes superpuestas, estos indican una pérdida de características a causa de las variaciones en el color.

- En la Figura 63, se presenta el ejemplo de una imagen con el nivel máximo de incremento de brillo ($\gamma = 4$). En este caso, ResNet50 y DenseNet121 son los modelos que mejor identifican al cuy, a pesar del mayor brillo en la imagen. Los mapas de calor revelan que el aumento uniforme de la intensidad de los píxeles no afecta significativamente la capacidad de los modelos para identificar al cuy, pero sí causa que las imágenes pierdan algunas zonas oscuras necesarias para segmentar al cuy de su entorno. Paradójicamente, el incremento del brillo en la imagen no afecta la capacidad de un observador humano para reconocer al cuy con la misma facilidad que en la imagen original.

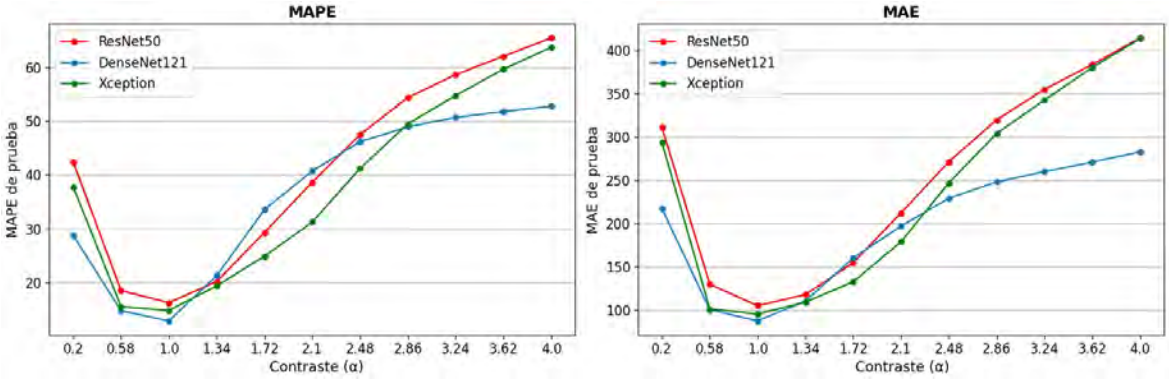
- De manera general, se puede afirmar que las redes experimentan un impacto moderado tanto en los niveles de oscurecimiento como en los de aclaramiento. Se considera moderado a este fenómeno debido a que el brillo provoca que todos los

valores de los píxeles (intensidades de cada canal de color RGB) de la imagen aumenten o disminuyan de manera simétrica, lo cual también afecta de manera simétrica a los filtros calculados por las redes neuronales.

Resultados Basados en el Contraste. Los resultados del rendimiento general de los modelos bajo variaciones de contraste se presentan en la Figura 64 a continuación:

Figura 64

Curvas de rendimiento de los modelos bajo distintos niveles de contraste



Nota. El nivel 1 indica la ausencia de la distorsión.

Sobre la distorsión:

- El comportamiento del contraste se manifiesta de manera análoga al brillo: cuando el factor α oscila entre los valores 0 y 1, oscureciendo la imagen (reducción de contraste), y cuando el factor α varía entre los valores 1 y 4, aclarando la imagen (incremento de contraste).
- En términos generales, las variaciones de contraste tienen un impacto significativo en los modelos, con el error promedio de predicción superando fácilmente el 50 % y los 275 gramos, y alcanzando hasta 400 gramos en niveles altos de incremento de contraste. Este impacto es notablemente mayor en comparación con otras distorsiones analizadas, como el brillo y el desenfoque de movimiento.
- En comparación con el brillo, el contraste impacta más en las arquitecturas durante los niveles de aclaramiento que en los niveles de oscurecimiento. Esto se explica por la diferencia en los niveles de error, que es aproximadamente 100 gramos.

Sobre los modelos:

- Globalmente y considerando el MAPE y MAE, en los niveles de reducción de contraste ($0 < \alpha < 1$), los tres modelos muestran comportamientos similares, pero en definitiva, DenseNet121 se ve menos afectado que los otros modelos.
- En términos de MAPE y durante los niveles de aclaramiento bajos ($1 < \alpha \leq 2{,}48$), se observa que DenseNet121 experimenta el mayor desajuste en sus predicciones, mientras que Xception obtiene mejores estadísticas de rendimiento, lo cual es un indicador de que Xception es el modelo más indicado cuando las imágenes presentan niveles bajos de incremento de contraste.
- En los niveles más altos del incremento de contraste ($2{,}48 < \alpha$), DenseNet121 logra el menor error de los tres modelos, manteniéndose estable en un 50 % y cerca de 300 gramos en promedio. Por otro lado, ResNet50 y Xception incrementan sus errores de manera similar hasta alcanzar el 60 % y más de 400 gramos en promedio.

Figura 65

Ejemplo del análisis de la reducción de contraste.

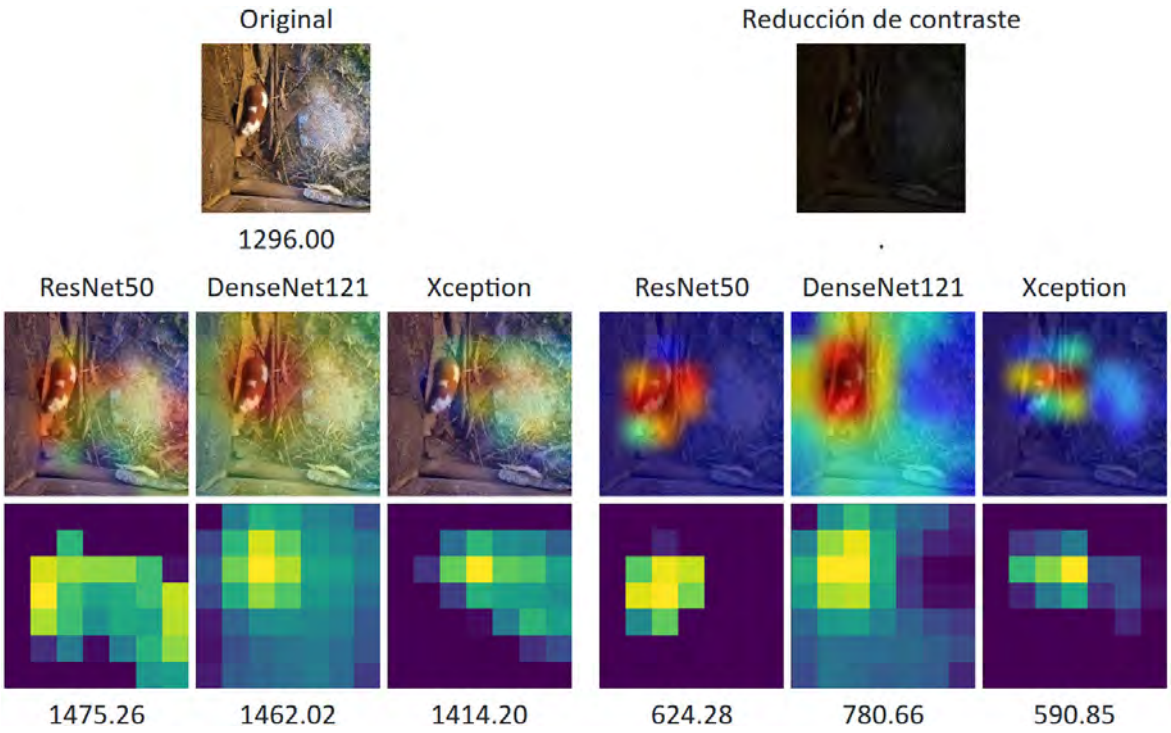
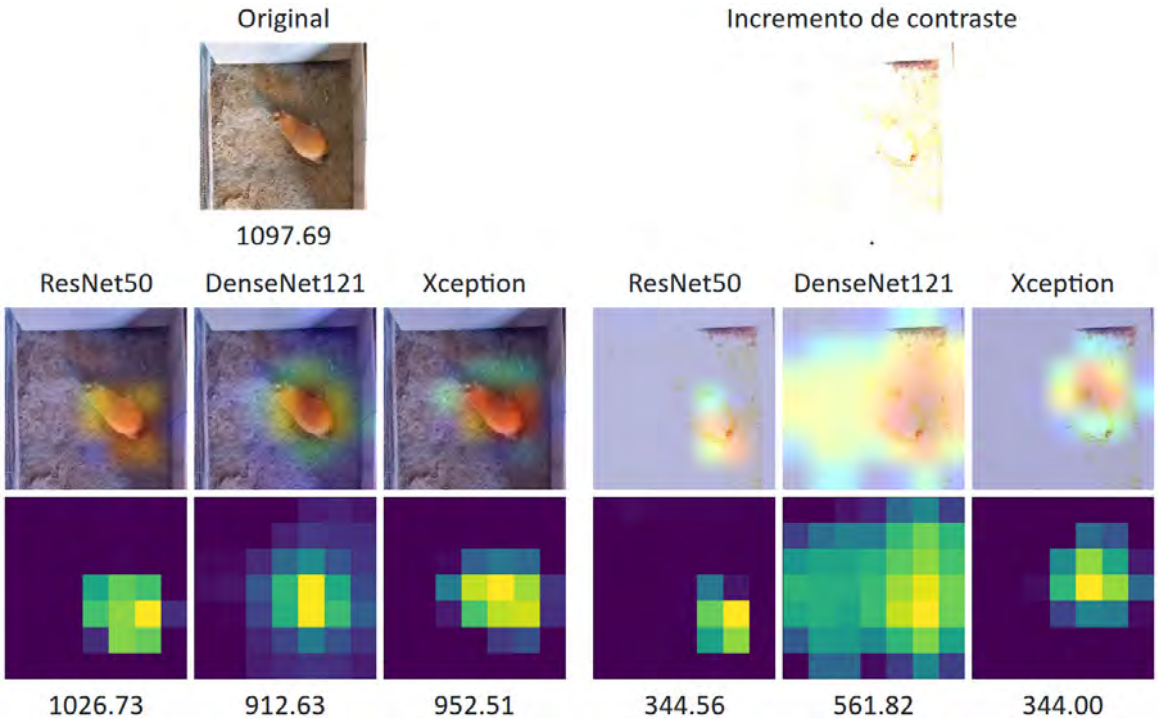


Figura 66

Ejemplo del análisis del incremento de contraste.



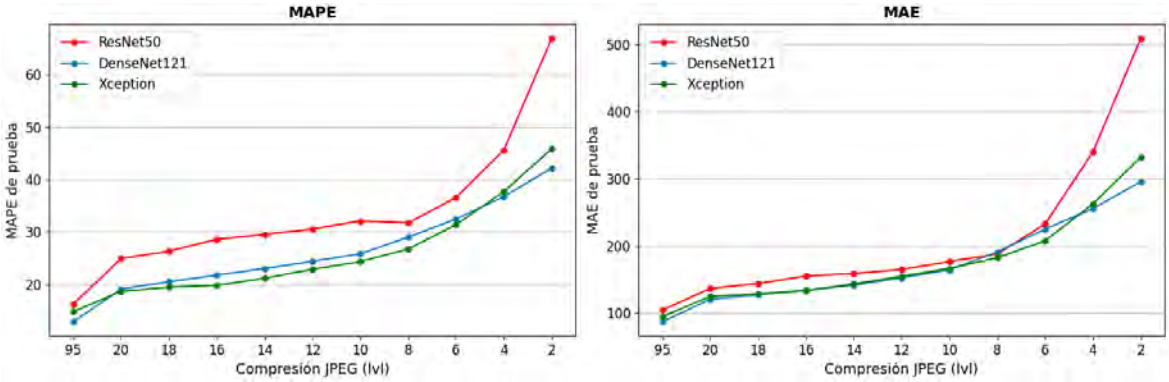
Algunas observaciones de las Figuras 65 y 66 son:

- En la Figura 65, que ilustra el grado máximo de reducción de contraste, se observa que la imagen se oscurece considerablemente, alterando los colores y dificultando a las redes distinguir con precisión al cuy de su entorno. Los mapas de calor revelan una notable pérdida de características esenciales, lo que resulta en una reducción significativa en el área identificada como perteneciente al cuy, y en consecuencia, en una disminución de la precisión de la predicción del peso.
- Por otro lado, en la Figura 66, que ilustra el grado máximo de incremento de contraste, los mapas de calor muestran que los modelos no logran detectar completamente el cuerpo del cuy debido a la supresión de los colores del cuy y de su entorno. Este fenómeno impacta directamente en la precisión de la detección del área ocupada por el animal.
- En resumen, el impacto del contraste es bastante significativo debido a la falta de diferenciación entre las intensidades de los píxeles (colores) oscuros y claros, lo que dificulta reconocer con precisión el área ocupada por el cuy.

Resultados Basados en la Compresión JPEG. Los resultados de los modelos frente a distintos niveles de compresión JPEG se presentan en la Figura 67.

Figura 67

Curvas de rendimiento de los modelos bajo distintos niveles de compresión JPEG



Nota. El nivel 95 indica la mínima compresión posible.

Sobre la distorsión:

- El "nivel de compresión JPEG" se corresponde de manera inversa con el parámetro lvl que representa la "calidad de la imagen". Por lo tanto, a cuanto mayor sea el valor de lvl, menor será el nivel de compresión JPEG.
- En líneas generales, la compresión JPEG no afecta drásticamente a los modelos en niveles bajos e intermedios de compresión. Sin embargo, a partir del nivel $lvl = 8$, los modelos muestran un aumento significativo en el error, evidenciado por un deterioro en la calidad de la predicción.

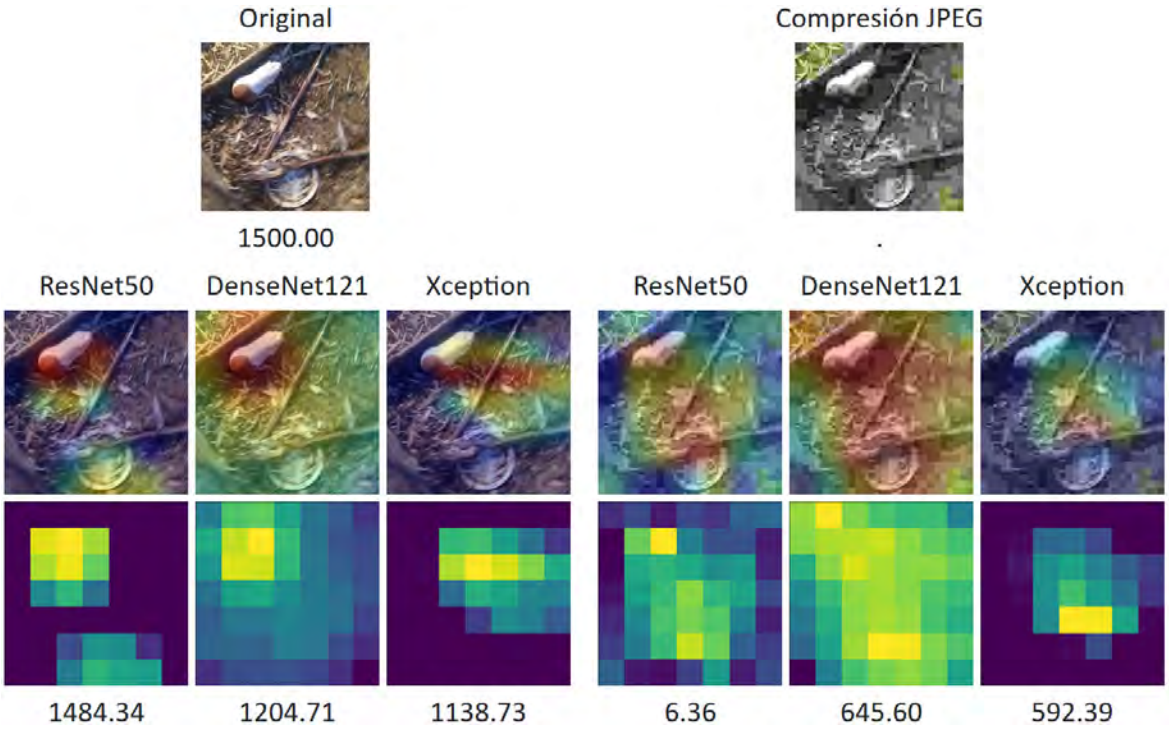
Sobre los modelos:

- A través de todos los niveles de compresión JPEG explorados y en términos de MAPE, los modelos DenseNet121 y Xception no se ven fuertemente afectados por la compresión JPEG, ya que no superan el 50 % de error. Sin embargo, ResNet50 es más sensible a esta distorsión, superando el 65 % de error. En cuanto a MAE, los errores de los tres modelos son más uniformes y no superan los 250 gramos hasta el nivel 6. Después de esto, ResNet50 experimenta un aumento brusco, superando los 500 gramos hacia el final.

- Las redes DenseNet121 y Xception demuestran ser altamente resistentes a los niveles de compresión bajos e intermedios ($lvl \geq 8$), manteniendo errores por debajo del 30 % y los 200 gramos. Por otro lado, ResNet50 no muestra la misma resistencia, ya que presenta un mayor error en todo este intervalo.
- Después del nivel $lvl = 8$, DenseNet121 y Xception exhiben una menor resistencia, mostrando errores por encima del 35 % y los 350 gramos hacia el final. Así también, ResNet50 se ve críticamente afectado, como se observa en la pronunciada pendiente de su curva.

Algunas evaluaciones visuales se presentan en las Figuras 68 y 69, y se realizan las observaciones posteriores:

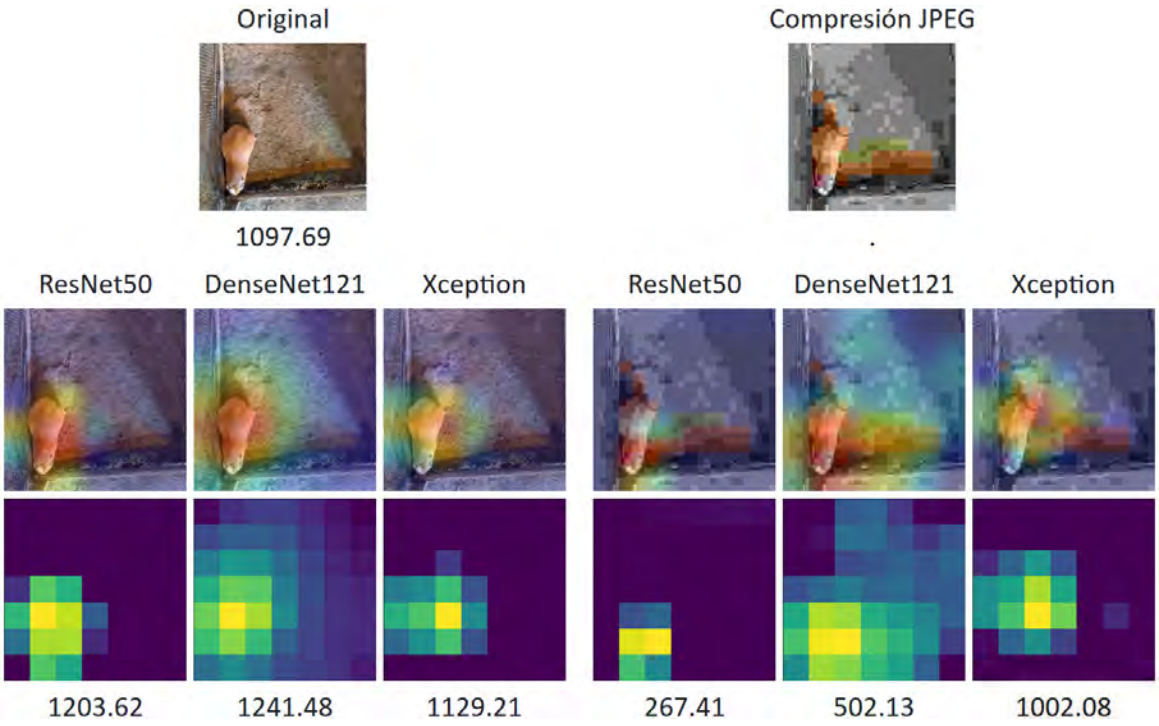
Figura 68
Primer ejemplo del análisis visual de la compresión JPEG.



- La compresión JPEG, que afecta principalmente a las crominancias (colores) de la imagen, no parece tener un efecto adverso considerable cuando las CNN intentan localizar al cuy dentro de la imagen, siempre y cuando el color del animal no se vea comprometido. Sin embargo, una compresión excesiva puede llevar a la pixelación

Figura 69

Segundo ejemplo del análisis visual de la compresión JPEG.



de la imagen y en consecuencia afectar el contorno del animal.

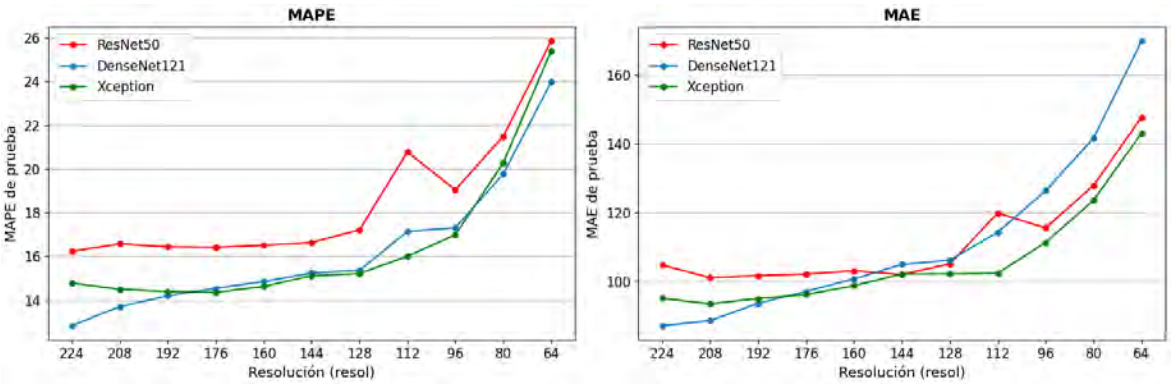
- En la Figura 68, donde los colores del animal se ven fuertemente comprometidos, las tres redes tienen dificultades para identificar al cuy. En este caso específico, solo DenseNet121 y Xception parecen tener éxito en esta tarea, pero a pesar de ello las predicciones poseen un error considerable.
- En la Figura 69, donde el color del animal se mantiene en cierta medida intacto, los tres modelos logran identificarlo correctamente. No obstante, la pixelación causada por la compresión hace que las redes consideren características diferentes que delimitan el contorno del cuy.
- En conclusión, la compresión JPEG afecta principalmente a los colores y al contorno del animal en las imágenes. Aunque los niveles bajos de compresión tienen un impacto mínimo, los niveles altos pueden afectar considerablemente la precisión de las predicciones. Esto plantea la cuestión de si el entrenamiento y la evaluación de los modelos utilizando imágenes en escala de grises podrían mejorar la resistencia de los modelos a la compresión JPEG, lo que podría ser un interesante

tema de investigación futura.

Resultados Basados en la Resolución. Los resultados de los modelos frente a distintos niveles de resolución se presentan en la Figura 70.

Figura 70

Curvas de rendimiento de los modelos bajo distintos niveles de resolución



Nota. El nivel 224 indica la ausencia de la distorsión.

Sobre la distorsión:

- Tras analizar el rendimiento de los modelos a diferentes niveles de reducción de resolución, se concluye que esta distorsión tiene un impacto mínimo en los modelos de CNN entrenados para la estimación del peso del cuy. Las curvas MAPE y MAE muestran una pendiente casi nula durante los niveles iniciales ($resol \leq 144$) para ResNet50 y Xception. Al comparar el nivel base ($resol = 224$) con la mínima resolución ($resol = 64$), el aumento en el error promedio es de aproximadamente un 10 % y 58 gramos.

Sobre los modelos:

- ResNet50 muestra una resistencia consistente a la reducción de resolución hasta el nivel $resol = 128$, tanto en las gráficas de MAE como en MAPE. Según estas observaciones, podría considerarse a esta red como una opción aceptable para trabajar con imágenes de baja resolución.
- Xception, en cambio, exhibe disminuciones mínimas en términos de MAPE y MAE durante los niveles iniciales. Lo que es un indicador de que la red puede mejorar su

rendimiento para este tipo de tareas al usar imágenes de menor resolución. Al analizar la inclinación total de su curva, se observa que no es pronunciada, lo que indica que en general la red es bastante resistente a esta distorsión.

- DenseNet121 también muestra una buena resistencia en los niveles iniciales, pero esta se ve reducida en los niveles finales. Por lo tanto, podríamos considerar que DenseNet121 se ve más afectada por la reducción de resolución que sus contrapartes.

- En general, los tres modelos se ven muy poco afectados por la reducción de la resolución. Solo al llegar a una resolución baja, como $resol = 112$, se evidencia un aumento significativo en los errores MAE y MAPE de los modelos. Sin embargo, este aumento, en comparación con otras distorsiones, es bajo y sugiere que para tareas de regresión con CNN no es imperativo utilizar altas resoluciones.

Figura 71

Primer ejemplo del análisis visual de la resolución.

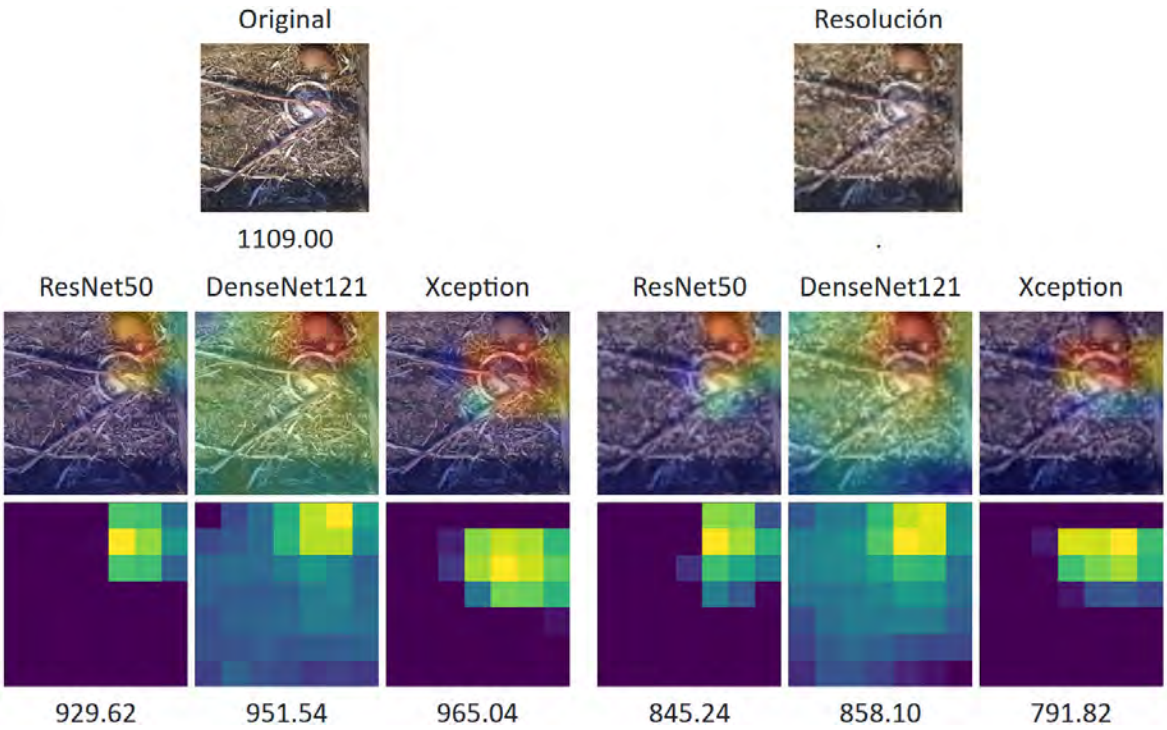
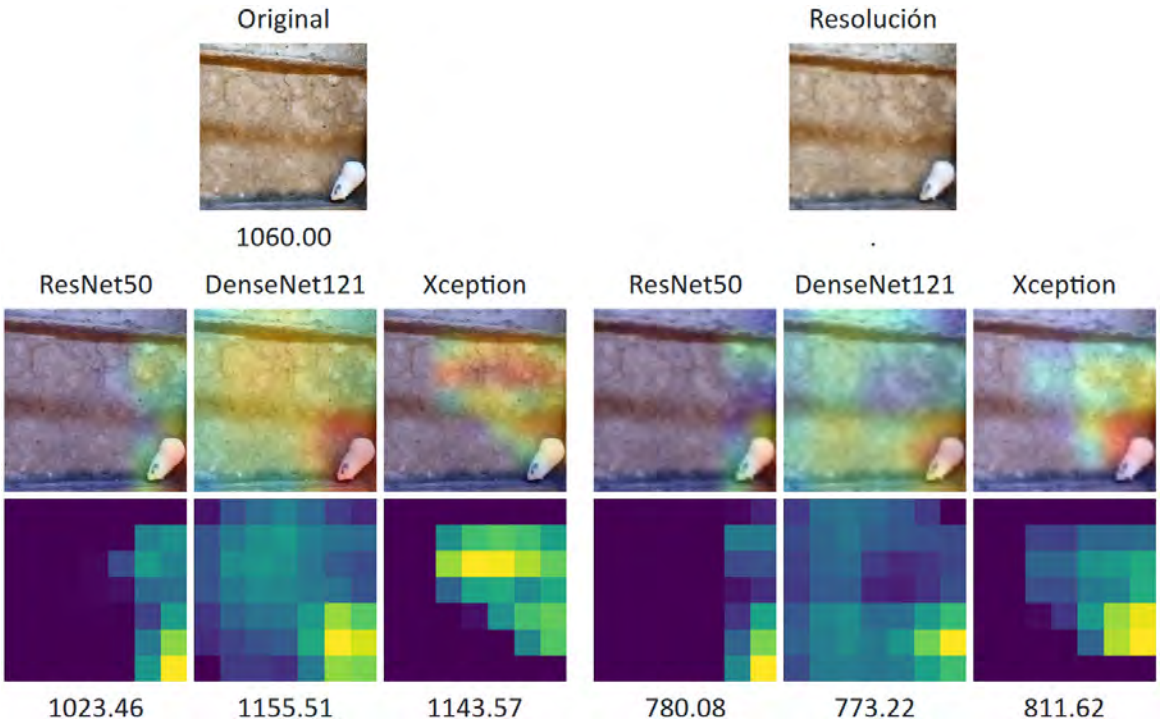


Figura 72

Segundo ejemplo del análisis visual de la resolución.



Algunas observaciones de las Figuras 71 y 72 son:

- La reducción de resolución provoca pixelación en la imagen, lo que afecta el contorno del animal y, por ende, incrementa el error en la predicción del peso. La pixelación dificulta la identificación precisa de las características del cuy, resultando en un aumento del error en la estimación del peso.
- En ambas figuras, los mapas de calor muestran que para ResNet50 y Xception solo hay pequeñas variaciones en las características consideradas. El principal indicador de la variabilidad del peso del cuy son las características que las redes toman en consideración, las cuales, como se observa en ambos ejemplos, varían sutilmente, indicando que las redes son bastante resistentes a esta distorsión.
- En contraste con las otras dos redes, DenseNet121 es significativamente más vulnerable a esta distorsión. Esto se debe a su estructura interconectada, que involucra la consideración de un mayor número de características para la predicción. Además, se sabe que la predicción de una red depende de un porcentaje específico de cada una de las características consideradas.

- En resumen, se puede decir que la reducción de resolución afecta en baja medida al contorno de los objetos en una imagen, lo que provoca que las redes no pierdan tanta precisión a la hora de predecir el peso del cuy.

En conclusión, se proporciona el Cuadro 19, que resume el impacto general de cada distorsión sobre cada uno de los modelos:

Cuadro 19

Resumen del impacto de las siete distorsiones sobre los tres modelos seleccionados

Distorsión	Característica afectada	ResNet50	DenseNet121	Xception
Desenfoque gaussiano	Texturas	Alto	Medio	Medio
Desenfoque de movimiento	Área	Medio	Medio	Medio
Ruido gaussiano	Contorno	Alto	Medio	Alto
Brillo	Colores	Medio	Bajo	Bajo
Contraste	Colores y área	Alto	Medio	Alto
Compresión JPEG	Colores y contorno	Alto	Medio	Medio
Resolución	Contorno	Bajo	Bajo	Bajo

Nota. Bajo (MAPE $\leq 30\,\%$), Medio ($30\,\% <$ MAPE $\leq 60\,\%$), Alto ($60\,\%$ <MAPE).

El mayor resultado obtenido en este estudio es a partir de la prueba cruzada en relación con la iluminación. Se descubrió que la luz ambiental, aunque representa un escenario complejo para los modelos durante el entrenamiento, es la opción más idónea para obtener el menor error posible en la predicción de los pesos de los cuyes. Este tipo de iluminación entrena de manera adecuada las capacidades de los modelos para captar las características más esenciales de los cuyes, como el contorno y el área del animal, facilitando su segmentación del entorno. Además, permite a los modelos reconocer correctamente los colores y las texturas del pelaje, evitando confusiones con otros objetos del entorno. Como resultado, se obtiene una predicción más precisa del peso del animal.

Aunque combinar varios tipos de iluminación contribuye a mejoras marginales en el rendimiento de los modelos, los resultados obtenidos bajo luz ambiental son significativamente notables. En particular, se observó que el modelo DenseNet121 es el más destacado, ya que presenta el menor error de predicción tanto en MAE como en MAPE en comparación con los otros modelos evaluados. Esto se debe a la arquitectura interconectada de DenseNet121, la cual emplea conexiones densas entre cada bloque convolucional, permitiendo el flujo de información entre todas las capas de la red. Esta

característica evita la pérdida de información crítica y mejora la propagación de gradientes durante el entrenamiento, minimizando el problema de la desaparición de gradiente en redes profundas. Además, la reutilización de características facilita que el modelo capture tanto patrones locales como globales, lo cual es fundamental para diferenciar las características específicas de los cuyes y ajustarse mejor a las variaciones de iluminación.

Discusión de Resultados Respecto a los Antecedentes

1. En el estudio de Dodge y Karam (2016), se analiza cómo la calidad de imagen afecta a las redes neuronales convolucionales (CNN) en tareas de clasificación, utilizando ejemplos contradictorios. De manera similar, nuestra investigación introduce distorsiones para simular la degradación de la calidad de imagen.

   Nuestros resultados indican que las redes son altamente sensibles al desenfoque, especialmente en relación con la pérdida de texturas, lo cual se alinea con los hallazgos sobre el desenfoque gaussiano en el trabajo de Dodge y Karam (2016). Asimismo, se observa una coincidencia en la alta sensibilidad de las CNN al ruido gaussiano, tal como se reportó en su investigación.

   Por otro lado, este estudio confirma la baja sensibilidad y alta resistencia de los modelos ante la compresión JPEG, mostrando una resistencia significativa que solo se ve comprometida en niveles extremos de compresión. En contraste, aunque Dodge y Karam (2016) encontraron una buena resistencia a cambios de contraste, nuestros resultados revelan una alta sensibilidad de los modelos al incremento del contraste. Esta discrepancia se debe a que, a diferencia del estudio mencionado, aquí se exploraron tanto la reducción como el aumento del contraste, lo que permitió observar un comportamiento diferente.

2. En nuestro estudio se evaluaron los efectos de diferentes tipos de iluminación, tanto ambiental como suplementaria, en la calidad de imagen, siguiendo el enfoque de Ranjan *et al.* (2023), que analizó su impacto en un modelo de detección de peces en un tanque subacuático. Sin embargo, a diferencia de su investigación, donde se concluyó que las condiciones de iluminación no influyen significativamente en el

rendimiento de los modelos, nuestros hallazgos indican que la luz ambiental es el tipo de iluminación que proporciona mayor robustez a los modelos.

Además, se observó que la combinación de diferentes tipos de iluminación puede resultar en modelos aún más robustos, sugiriendo que la estrategia de iluminación es un factor crucial a considerar para optimizar el rendimiento de las redes neuronales.

3. El estudio de Hu *et al.* (2021), que investigó la influencia de la calidad de imagen y la consistencia de la iluminación en la detección y segmentación de hierbas, sugiere que las imágenes recolectadas en condiciones de iluminación deficiente pueden proporcionar un buen rendimiento de las CNN para el mapeo de hierbas. Estos resultados coinciden con nuestras observaciones sobre la luz ambiental, que demuestran ser la opción más adecuada para entrenar modelos robustos en la estimación del peso de los cuyes.

   En cuanto al desenfoque de movimiento, nuestro estudio determinó que esta distorsión afecta de manera moderada el rendimiento de las redes. En contraste, Hu *et al.* (2021) encontraron un efecto mínimo del desenfoque de movimiento en las tres redes evaluadas en su trabajo, lo que podría explicarse por la diferencia en el tamaño del kernel utilizado $ks = 9$ frente al $ks = 12$ en nuestra investigación.

   Respecto al ruido gaussiano, nuestros hallazgos indican que esta distorsión afecta significativamente el rendimiento de los modelos en la estimación del peso. Sin embargo, Hu *et al.* (2021) reportaron efectos mínimos en los niveles iniciales de distorsión, posiblemente debido a las diferencias en el rango de valores utilizados ($80 \leq \sigma \leq 640$) y el tamaño de imagen (2048×2048 px) empleado en su estudio.

   Finalmente, nuestros resultados muestran que la reducción de resolución tiene un impacto mínimo en los modelos de CNN entrenados para la estimación del peso de los cuyes. Este resultado contrasta con el estudio de Hu *et al.* (2021), donde se observó una mayor sensibilidad a la reducción de resolución. Esta discrepancia podría atribuirse a la naturaleza de las tareas evaluadas, ya que la detección y

segmentación requieren una delimitación precisa de los objetos, a diferencia de la estimación de peso, que en general parece prescindir de tal precisión.

4. Los resultados de Akkoca Gazioğlu y Kamaşak (2021), en su estudio sobre la clasificación de melanoma, indican que el contraste afecta más a las arquitecturas durante los niveles de incremento que en los de reducción de contraste, tanto para imágenes de lesiones benignas como de melanoma. De manera similar, en esta investigación se observó que los modelos es más sensible a los niveles de incremento de contraste, replicando el comportamiento descrito en su estudio.

   Asimismo, al igual que en la investigación de Akkoca Gazioğlu y Kamaşak (2021), el modelo DenseNet121 demostró ser el más robusto frente a la mayoría de las distorsiones y variaciones de iluminación evaluadas. Estos resultados no solo corroboran la estabilidad y capacidad de adaptación de esta arquitectura en diferentes dominios, sino que también destacan la relevancia de seguir investigando su desempeño en contextos donde las variaciones en la calidad de imagen son significativas.

5. En el estudio de Zapata-Ttito (2022), el mejor rendimiento del modelo CNN para la predicción del peso del cuy se obtuvo con la variante ResNet50A, alcanzando un MAPE de 12.41 % en 150 épocas. En comparación, nuestro modelo DenseNet121 mostró un MAPE de 12.84 %, siendo el resultado más cercano al de su investigación. Sin embargo, lo destacable es que nuestro modelo logró este rendimiento en solo 35 épocas, lo que evidencia la eficiencia y el alto desempeño de DenseNet121 para esta tarea, reduciendo considerablemente el tiempo de entrenamiento sin sacrificar precisión.

Conclusiones

1. La calidad de la imagen, influenciada por variaciones en la iluminación y distorsiones, impacta significativamente en el rendimiento de las CNN entrenadas para la estimación del peso del cuy. Sin embargo, es importante señalar que las CNN muestran diferentes niveles de sensibilidad a cada factor y distorsión. La iluminación es un factor determinante que afecta la capacidad de los modelos para identificar características clave de las imágenes. Por otro lado, las distorsiones afectan elementos fundamentales de la imagen, como colores, contornos, texturas y el área del cuy, lo que compromete su capacidad predictiva, según lo indicado por las métricas de error utilizadas para evaluar estas redes de regresión. Finalmente, al comparar los modelos, DenseNet121 fue la red con mejor rendimiento para estimar el peso del cuy, destacándose tanto ante variaciones en la iluminación como frente a las distorsiones analizadas.

2. Se determinaron siete distorsiones relevantes que afectan el rendimiento de las CNN entrenadas para la estimación del peso corporal del cuy. Estas distorsiones incluyen desenfoque gaussiano, desenfoque de movimiento, ruido gaussiano, brillo, contraste, compresión JPEG y resolución.

3. Se logró reunir un conjunto significativo de 3000 imágenes de 118 cuyes, distribuidas de manera equitativa entre los conjuntos de datos de iluminación ambiental y suplementaria. Además, se construyó un dataset principal con 4561 imágenes vinculadas a los pesos de 343 cuyes, aumentando posteriormente su tamaño a 18244 imágenes mediante técnicas de aumento de datos. Este dataset se dividió eficazmente en conjuntos de entrenamiento, validación y prueba. A partir de este último se crearon datasets de prueba individuales, cada uno afectado por una distorsión específica.

4. Al comparar los tres modelos seleccionados, se observó que DenseNet121 ofreció el mejor desempeño tanto ante variaciones en la iluminación como frente a la mayoría de las distorsiones analizadas, con un impacto en su rendimiento, medido en términos de MAPE, inferior al 60 %. Xception se destacó como la segunda red más resistente, siendo críticamente afectada solo por el ruido gaussiano y la variación en el contraste. Por último, ResNet50 resultó ser la más vulnerable frente a la mayoría de las distorsiones evaluadas, mostrando un MAPE superior al 60 % en estos casos y demostrando resiliencia únicamente ante la reducción de resolución.

Recomendaciones y Trabajos Futuros

- Para minimizar los efectos y prevenir la aparición de las distorsiones evaluadas, se recomienda utilizar cámaras de alta calidad, correctamente calibradas y estabilizadas, además de ajustar adecuadamente los parámetros de captura y mantener la lente limpia y en óptimas condiciones.
- Otra recomendación importante es controlar la iluminación del entorno, garantizando que sea uniforme y adecuada para evitar sombras excesivas y asegurar un brillo y contraste consistentes en las imágenes.
- Asimismo, se sugiere evitar la compresión durante la captura, configurando la cámara para trabajar en formatos de alta calidad como RAW, a fin de evitar la introducción de artefactos y la pérdida de información. Del mismo modo, se debe emplear una resolución nativa y óptima para preservar la calidad original de la imagen.
- Es recomendable ampliar el dataset de cuyes, incrementando tanto la diversidad de pesos como la cantidad de imágenes por ejemplar, lo que resultará en una mejora en la capacidad predictiva de las Redes Neuronales Convolucionales.
- Se propone continuar investigando la arquitectura DenseNet, dada su prometedora aplicabilidad en diversas tareas, además de seguir explorando posibles ajustes o mejoras en su implementación.
- Se plantea la necesidad de analizar cómo la variación en la distancia entre la cámara y el cuy impacta el rendimiento de los modelos de CNN, lo que podría proporcionar revelaciones valiosas sobre la influencia de este factor en la precisión de la estimación de peso.

- Se sugiere estudiar la eficacia comparativa entre el incremento de ejemplares de cuy (mayor diversidad de pesos) y el aumento en la cantidad de imágenes de cada cuy, para determinar la estrategia más efectiva en la mejora del desempeño de los modelos.
- Se propone llevar a cabo una investigación para evaluar la necesidad real de los tres canales de colores en la estimación del peso animal, explorando la posibilidad de trabajar con imágenes en escala de grises y sus posibles beneficios.

Bibliografía

Agencia-Peruana-de-Noticias (2022). Carne de cuy: estas son las bondades nutricionales de este alimento ancestral andino. `http://surl.li/gfjgy`. Accessed: 2023-4-7.

Akkoca Gazioğlu, B. S. y Kamaşak, M. E. (2021). Effects of objects and image quality on melanoma classification using deep neural networks. *Biomed. Signal Process. Control*, 67(102530):102530.

Alcázar, J. A. (2014). Explicación del sistema de compresión JPEG.

Aqqa, M., Mantini, P., y Shah, S. (2019). Understanding how video quality affects object detection algorithms. En *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications.

Bianco, S., Cadene, R., Celona, L., y Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6:64270–64277.

Boureau, Y.-L., Ponce, J., y Lecun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. pp. 111–118.

Buayai, P., Piewthongngam, K., Leung, C., y Runapongsa Saikaew, K. (2019). Semi-automatic pig weight estimation using digital image analysis. *Applied Engineering in Agriculture*, 35:521–534.

Chauca, L. (1997). *Produccion de Cuyes (Cavia Porcellus)*.

Chen, R., Zhao, Y., Yang, Y., Wang, S., Li, L., Sha, X., Liu, L., Zhang, G., y Li, W. J. (2023). Online estimating weight of white pekin duck carcass by computer vision. *Poult. Sci.*, 102(2):102348.

Chollet, F. (2015). keras. `https://keras.io/api/applications`.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions.

Crane, R. (1997). *A Simplified Approach to Image Processing: Classical and Modern Techniques in C*. HP Professional Series. Prentice Hall PTR.

Dodge, S. F. y Karam, L. J. (2016). Understanding how image quality affects deep neural networks. *CoRR*, abs/1604.04004.

Engeldrum, P. G. (2004). A short image quality model taxonomy. *Journal of Imaging Science and Technology*, 48:160–165.

Fernandes, A. F. A., Turra, E. M., de Alvarenga, É. R., Passafaro, T. L., Lopes, F. B., Alves, G. F. O., Singh, V., y Rosa, G. J. M. (2020). Deep learning image segmentation for extraction of fish body measurements and prediction of body weight and carcass traits in nile tilapia. *Comput. Electron. Agric.*, 170(105274):105274.

Food and Agriculture Organization (s.f.). Producción de cuyes (cavia porcellus). `https://www.fao.org/3/W6562s/w6562s01.htm`. Accessed: 2023-4-7.

Gjergji, M., Weber, V., Silva, L., Gomes, R., Araujo, T., Pistori, H., y Alvarez, M. (2020). Deep learning techniques for beef cattle body weight prediction. pp. 1–8.

González, A., Martínez, F, Pernía, A, Alba, F., Castejón, M., Ordieres, J., y Vergara, E. (2006). *Técnicas y algoritmos básicos de visión artificial*. Universidad de la Rioja - Servicio de publicaciones.

Gonzalez, R. C. y Woods, R. E. (2018). *Digital Image Processing*. Pearson, 330 Hudson Street, New York, NY 10013.

Goodfellow, I., Bengio, Y., y Courville, A. (2016). *Deep Learning*. MIT Press.

Hansen, M. F., Smith, M. L., Smith, L. N., Abdul Jabbar, K., y Forbes, D. (2018). Automated monitoring of dairy cow body condition, mobility and weight using a single 3D video capture device. *Comput. Ind.*, 98:14–22.

He, K., Zhang, X., Ren, S., y Sun, J. (2015). Deep residual learning for image recognition.

Hernández-Sampieri, R., Fernández-Collado, C., y Baptista-Lucio, M. D. P. (2014). *Metodología de la investigación (6ta ed.)*. McGRAW-HILL, México D.F.

Hsiao, T.-Y., Chang, Y.-C., Chou, H.-H., y Chiu, C.-T. (2019). Filter-based deep-compression with global average pooling for convolutional networks. *J. Syst. Arch.*, 95:9–18.

Hu, C., Sapkota, B. B., Thomasson, J. A., y Bagavathiannan, M. V. (2021). Influence of image quality and light consistency on the performance of convolutional neural networks for weed mapping. *Remote Sens. (Basel)*, 13(11):2140.

Huang, G., Liu, Z., van der Maaten, L., y Weinberger, K. Q. (2018). Densely connected convolutional networks.

IBM (s.f.a). What are convolutional neural networks? `https://www.ibm.com/topics/convolutional-neural-networks`. Accessed: 2023-10-10.

IBM (s.f.b). What is computer vision? `https://www.ibm.com/topics/computer-vision`. Accessed: 2023-4-7.

Jeelani, H., Martin, J., Vasquez, F., Salerno, M., y Weller, D. (2018). Image quality affects deep learning reconstruction of MRI. En *IEEE*, pp. 357–360.

Jones, T. (2022). What is deep learning? definition, techniques, and use cases. `https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-deep-learning/`. Accessed: 2023-4-8.

Jun, K., Kim, S. J., y Ji, H. W. (2018). Estimating pig weights from images without constraint on posture and illumination. *Comput. Electron. Agric.*, 153:169–176.

Kanade, V. (2022). What is machine learning? definition, types, applications, and trends for 2022. `https://www.spiceworks.com/tech/`

artificial-intelligence/articles/what-is-ml/. Accessed: 2023-4-7.

Kashiha, M., Bahr, C., Ott, S., Moons, C. P. H., Niewold, T. A., Ödberg, F. O., y Berckmans, D. (2014). Automatic weight estimation of individual pigs using image analysis. *Comput. Electron. Agric.*, 107:38–44.

Kc, K., Yin, Z., Li, D., y Wu, Z. (2021). Impacts of background removal on convolutional neural networks for plant disease classification in-situ. *Agriculture*, 11(9):827.

Kingma, D. P. y Ba, J. (2017). Adam: A method for stochastic optimization.

Konovalov, D. A., Saleh, A., Efremova, D. B., Domingos, J. A., y Jerry, D. R. (2019). Automatic weight estimation of harvested fish from images.

López-Ramos, D. y Arco-García, L. (2019). Deep learning for aspect extraction in textual opinions. 13:105–145.

Meckbach, C., Tiesmeyer, V., y Traulsen, I. (2021). A promising approach towards precise animal weight monitoring using convolutional neural networks. *Comput. Electron. Agric.*, 183(106056):106056.

Ministerio de Desarrollo Agrario y Riego (2023). Cadena productiva de cuy.

Nixon, M. S. y Aguado, A. S. (2002). *Feature Extraction and Image Processing*. Newnes, Oxford, Inglaterra.

Pan, S. J. y Yang, Q. (2010). A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

Qiao, Y., Truman, M., y Sukkarieh, S. (2019). Cattle segmentation and contour extraction based on mask R-CNN for precision livestock farming. *Comput. Electron. Agric.*, 165(104958):104958.

Ranjan, R., Sharrer, K., Tsukuda, S., y Good, C. (2023). Effects of image data quality on a convolutional neural network trained in-tank fish detection model for recirculating aquaculture systems. *Comput. Electron. Agric.*, 205(107644):107644.

Real Academia Española (s.f.). *Distorsión*. Diccionario de la lengua española, 23.ª ed., [versión 23.6 en línea].

Sant'Ana, D. A., Pache, M. C. B., Martins, J., Soares, W. P., de Melo, S. L. N., Garcia, V., de Moares Weber, V. A., da Silva Heimbach, N., Mateus, R. G., y Pistori, H. (2021). Weighing live sheep using computer vision techniques and regression machine learning. *Mach. Learn. Appl.*, 5(100076):100076.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., y Batra, D. (2019). Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359.

Senthil Pandi, S., Senthilselvi, A., Gitanjali, J., ArivuSelvan, K., Gopal, J., y Vellingiri, J. (2022). Rice plant disease classification using dilated convolutional neural network with global average pooling. *Ecol. Modell.*, 474(110166):110166.

Shapiro, L. y StockMan, G. (2000). *Computer Vision*. Pearson, Upper Saddle River, NJ, Estados Unidos de América.

Sharma, S., Sharma, S., y Athaiya, A. (2020). Activation functions in neural networks. *International Journal of Engineering Applied Sciences and Technology*, 4(2455-2143):310–316.

Shorten, C. y Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *J. Big Data*, 6(1).

Shustanov, A. y Yakimov, P. (2019). Modification of single-purpose cnn for creating multi-purpose cnn. *Journal of Physics: Conference Series*, 1368:052036.

Simonyan, K. y Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Singh, P., Chaudhury, S., y Panigrahi, B. (2021). Hybrid mpso-cnn: Multi-level particle swarm optimized hyperparameters of convolutional neural network. *Swarm and Evolutionary Computation*, 63:100863.

Sonka, M., Hlavac, V., y Boyle, R. (2013). *Image Processing, Analysis, and Machine Vision*. Wadsworth Publishing, Belmont, CA, 4 edición.

Sucar, L. E. y Gómez, G. (2011). *Visión Computacional*. Instituto Nacional de Astrofísica, Óptica y Electrónica / Helmholtz Zentrum Munchen.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., y Rabinovich, A. (2014). Going deeper with convolutions.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., y Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312.

Thung, K.-H. y Raveendran, P. (2009). A survey of image quality measures. pp. 1–4.

Tiwari, S., Shukla, V. P., Singh, A. K., y Biradar, S. R. (2013). Review of motion blur estimation techniques. *Journal of Image and Graphics*, 1:176–184.

Uberoi, A. (2019). Motion blur in python. `https://www.geeksforgeeks.org/opencv-motion-blur-in-python/`. Accessed: 2023-10-16.

Wang, Y., Yang, W., Winter, P., y Walker, L. (2008). Walk-through weighing of pigs using machine vision and an artificial neural network. *Biosyst. Eng.*, 100(1):117–125.

Wang, Z., Bovik, A. C., y Lu, L. (2002). Why is image quality assessment so difficult? En *IEEE International Conference on Acoustics Speech and Signal Processing*. IEEE.

Wueller, D. y Kejser, U. B. (2016). Standardization of image quality analysis – ISO 19264. *Archiving*, 13(1):111–116.

Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A. K., y Almotairi, S. (2022). A comparison of pooling methods for convolutional neural networks. *Appl. Sci. (Basel)*, 12(17):8643.

Zapata-Ttito, A. G. (2022). Aplicación de visión artificial en la estimación del peso corporal del cuy. Repositorio Institucional UNSAAC, Cusco.

Zhang, W., Li, C., Peng, G., Chen, Y., y Zhang, Z. (2018). A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load. *Mechanical Systems and Signal Processing*, 100:439–453.

Apéndice A. Recursos adicionales

Enlaces al dataset principal y archivos de modelos entrenados

- Dataset principal
  https://drive.google.com/drive/folders/1dAcMlRIG-GGPj7Twrpz17LMtIZeqwG5Q?usp=drive_link
- Archivos de los modelos entrenados
  https://drive.google.com/drive/folders/1dHzXcP6N4rPpHC4ZInTRjx3Rf63rJ-nJ?usp=sharing

Enlaces a los cuadernos de entrenamiento

- Pruebas de iluminación
  https://drive.google.com/drive/folders/1Z3AVayCXfHfGZuow3Fy1rOi6Rd1wUQ3J?usp=drive_link
- Entrenamiento del dataset principal
  https://drive.google.com/drive/folders/1Z1L2nhTUdUh-vPhZ7jBoFstVpTdJzDH7?usp=drive_link
- Pruebas de distorsiones
  https://drive.google.com/drive/folders/1Z4jMTNpY2e3rFbA8WqalMTAH1GVgVaY8?usp=drive_link
- Pruebas de GradCAM
  https://drive.google.com/drive/folders/1dgWOb_FhN44B0XO2XKzrxeoAwx9dnEn7?usp=drive_link

Apéndice B. Descripción del dataset

Listing 9
Descripción del dataset

```
#### Nombre del Dataset:
Dataset de Imágenes de Cuyes para Estimación de Peso.
#### Descripción:
Este dataset contiene 4561 imágenes de cuyes capturadas en diferentes
 ↪  condiciones de iluminación (luz ambiental y luz suplementaria) para
 ↪  evaluar modelos de estimación de peso basados en redes neuronales
 ↪  convolucionales. Adicionalmente contiene los scripts para la
 ↪  introducción de distorsiones en el conjunto de prueba.
#### Fuente:
Las imágenes fueron capturadas en granjas locales de la región Cusco con
 ↪  cámaras RGB en condiciones controladas. El dataset es una combinación
 ↪  de las imágenes obtenidas para las pruebas de iluminación e imágenes
 ↪  del trabajo de [Zapata Ttito (2023)]
(http://repositorio.unsaac.edu.pe/handle/20.500.12918/7067).
#### Estructura:
- Total de Imágenes: 4561
- Formato: `.jpg`
- Subconjuntos:
    - Entrenamiento (`train`): 3649 imágenes
    - Validación (`val`) : 457 imágenes
    - Prueba (`test`) : 455 imágenes
#### Variables:
- `filename`: Nombre del archivo de la imagen (string).
    - **prefijos**: Indica el origen y naturaleza de las imágenes.
        - `B_`: Las imágenes son de luz ambiental.
        - `CUY_00`: Las imágenes pertenecen al trabajo de [Zapata Ttito
         ↪  (2023)]
        (http://repositorio.unsaac.edu.pe/handle/20.500.12918/7067).
- `weight`: Peso del cuy en gramos (numérico).
#### Limitaciones:
El dataset está limitado a cuyes de tipo de pelaje lacio, crespo y
 ↪  ensortijado. No se tuvo acceso a cuyes con pelaje landoso. Esta
 ↪  clasificación se puede encontrar en el trabajo de [Ataucusi (2015)]
(https://www.academia.edu/31831975/MANEJO_T%C3%89CNICO_DE_LA_
CRIANZA_DE_CUYES).
#### Licencia:
Este dataset está disponible bajo la licencia **Creative Commons
 ↪  Atribución-No Comercial-Compartir Igual 4.0 Internacional (CC
 ↪  BY-NC-SA 4.0)**. Puede utilizar, redistribuir y modificar el dataset
 ↪  bajo los siguientes términos:
Puede ver una copia completa de la licencia en el siguiente enlace:
https://creativecommons.org/licenses/by-nc-sa/4.0/.
```