

# Trabajo Final Integrador

## Visión por Computadora II

## Dataset TrashNet

### INTEGRANTES:

- Diego Araujo
- Christopher Charaf
- Azul Villanueva



# Tabla de contenidos

**01**

**Problema y  
dataset  
TrashNet**

**02**

**Data  
augmentation**

**03**

**Modelos  
empleados**

**04**

**Análisis y  
resultados  
obtenidos**



# Tabla de contenidos

**05**

**Optimización y  
ajuste de  
hiperparámetros**

**06**

**Conclusiones y  
sugerencias**



# Conjunto de Datos TrashNet

El problema que se aborda con este dataset es la clasificación de residuos para reciclaje.

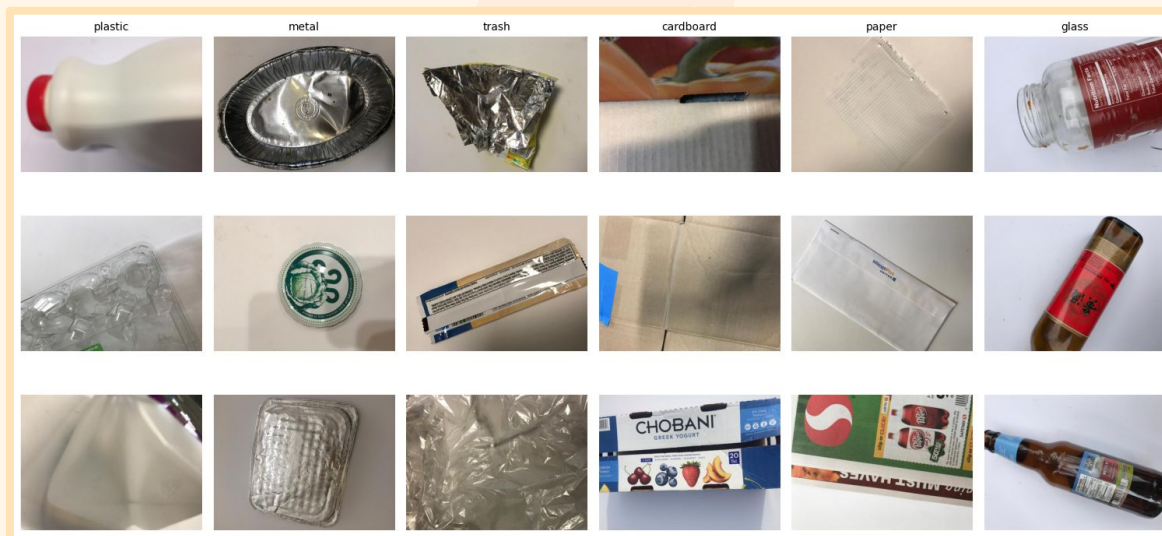
Para resolverlo, se entrena un modelo que, a partir de imágenes, pueda identificar y clasificar distintos tipos de residuos (plástico, vidrio, papel y orgánicos), con el objetivo de optimizar los procesos de reciclaje y la gestión de residuos.

# Conjunto de Datos TrashNet

El dataset está compuesto por carpetas que contienen imágenes de residuos organizadas en distintas categorías. Cada categoría incluye mayormente entre 400 y 500 imágenes en formato JPG, con una resolución de 512x384 píxeles.

## Distribución por clase:

- **Basura:** 137 imágenes
- **Papel:** 594 imágenes
- **Metal:** 410 imágenes
- **Plástico:** 482 imágenes
- **Vidrio:** 501 imágenes
- **Cartón:** 403 imágenes



# Conjunto de Datos TrashNet

Este problema de clasificación presenta dos desafíos clave:

- **Escasez de datos, junto con un desbalance en las clases.**
- **Alta semejanza visual entre imágenes de distintas categorías, lo que puede dificultar la correcta clasificación.**

Este último punto puede observarse en los ejemplos mostrados a la derecha.



# Data augmentation

El dataset presenta **variaciones limitadas en fondo, iluminación y ángulo**, lo que puede llevar al modelo a aprender patrones específicos en lugar de generalizar correctamente.

Para mitigar este problema, se aplicaron **técnicas de *Data Augmentation*** (rotaciones, variaciones, distorsiones, etc) con el objetivo de **simular condiciones más realistas** de descarte de residuos.

# Data augmentation

Para aumentar la diversidad visual del dataset se realizaron las siguientes transformaciones:

- **Rotaciones**, para representar residuos en distintas orientaciones.
- **Volteos horizontales y verticales**, simulando cómo pueden ser depositados.
- **Zoom y escalado**, para reflejar diferentes tamaños y distancias respecto a la cámara.
- **Ajustes de color y brillo**, imitando condiciones de iluminación variables (por ejemplo, interior vs. exterior).

Estas técnicas ayudaron a **reducir el *overfitting*** en los modelos entrenados, al introducir variaciones visuales sin necesidad de recolectar más datos.



# Data augmentation (implementación)

```
# Data augmentation para training
train_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomRotation(30),
    transforms.RandomHorizontalFlip(),
    transforms.RandomVerticalFlip(),
    transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1),
    transforms.RandomAffine(degrees=0, translate=(0.1, 0.1), scale=(0.9, 1.1)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

# Transformaciones básicas para validation
val_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])
```

## Ejemplos luego de la transformación



# Modelos empleados – CNN Convencional

## Arquitectura de la Red Neuronal (CNN)

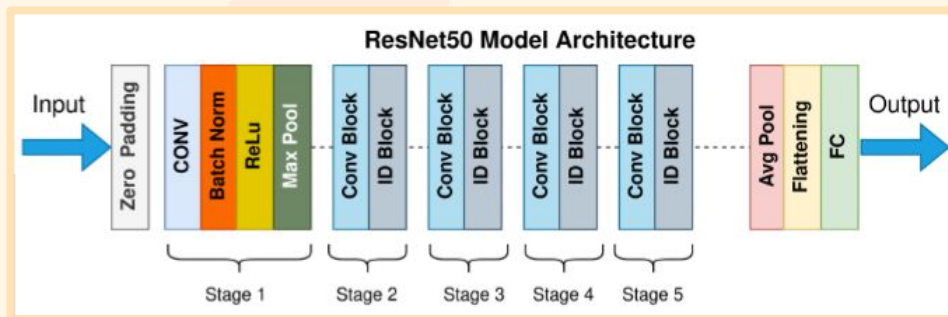
- **Entrada:** Imágenes de tamaño 224x224 píxeles, con 3 canales (RGB).
- **Capas Convolucionales:**  
4 bloques compuestos por:
  - *Conv2D* + *ReLU* con 32, 64, 128 y 256 filtros
  - *MaxPooling2D* para reducción de dimensiones
- **Capas Densas (Fully Connected):**
  - *Flatten* implícito para vectorizar las características
  - Capa oculta con **512 neuronas**, activación *ReLU* y *Dropout* para regularización
  - Capa de salida con **6 neuronas** (una por clase)
- **Parámetros entrenables:** ~ 26 millones

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 224, 224]	896
ReLU-2	[-1, 32, 224, 224]	0
MaxPool2d-3	[-1, 32, 112, 112]	0
Conv2d-4	[-1, 64, 112, 112]	18,496
ReLU-5	[-1, 64, 112, 112]	0
MaxPool2d-6	[-1, 64, 56, 56]	0
Conv2d-7	[-1, 128, 56, 56]	73,856
ReLU-8	[-1, 128, 56, 56]	0
MaxPool2d-9	[-1, 128, 28, 28]	0
Conv2d-10	[-1, 256, 28, 28]	295,168
ReLU-11	[-1, 256, 28, 28]	0
MaxPool2d-12	[-1, 256, 14, 14]	0
Linear-13	[-1, 512]	25,690,624
ReLU-14	[-1, 512]	0
Dropout-15	[-1, 512]	0
Linear-16	[-1, 6]	3,078
Total params: 26,082,118		
Trainable params: 26,082,118		
Non-trainable params: 0		
Input size (MB): 0.57		
Forward/backward pass size (MB): 51.69		
Params size (MB): 99.50		
Estimated Total Size (MB): 151.76		

# Modelos empleados – ResNet50

## Arquitectura de la Red Neuronal (ResNet50)

- **Entrada:** imágenes de tamaño 224x224 píxeles, con 3 canales (RGB).
- **Preprocesamiento:**
  - Capa de entrada con **padding cero** (Zero Padding)
  - Procesamiento inicial (Shape 1-5)
- **Bloques Convolucionales:**  
4 bloques compuestos por:
  - Capa **CONV** (Convolución)
  - *Batch Normalization* (Normalización)
  - *ReLU* (Función de activación)
  - *MaxPooling* (Reducción dimensional)
- **Capas Finales:**
  - *Average Pooling* (Agrupamiento promedio)
  - *Flattening* (Aplanamiento)
  - *Fully Connected* (Capa densa final)
  - Salida (Output)
- **Parámetros entrenables:** ~25 millones



# Modelos empleados – ResNet50

## Características de Resnet50:

- **Arquitectura profunda con 50 capas** (de ahí el nombre ResNet50).
- Uso de "skip connections" (conexiones residuales) que permiten entrenar redes muy profundas.
- Bloques de identidad (ID Block) que mantienen la dimensión.
- Bloques convolucionales (Conv Block) que modifican la dimensión

## Ventajas:

- Eficaz para problemas complejos de visión por computadora.
- Menor problema de vanishing gradients que otras arquitecturas profundas.
- Buen desempeño en clasificación de imágenes.

# Análisis y resultados obtenidos

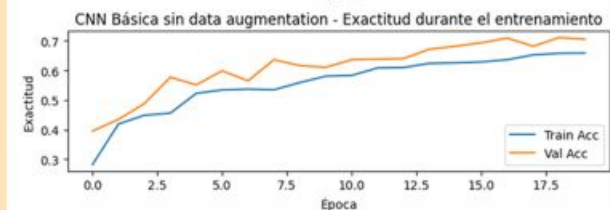
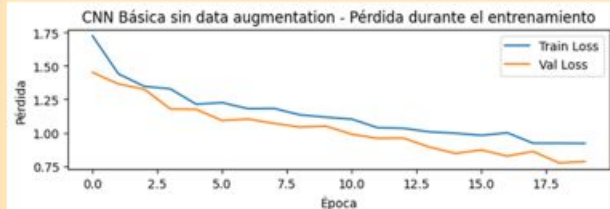
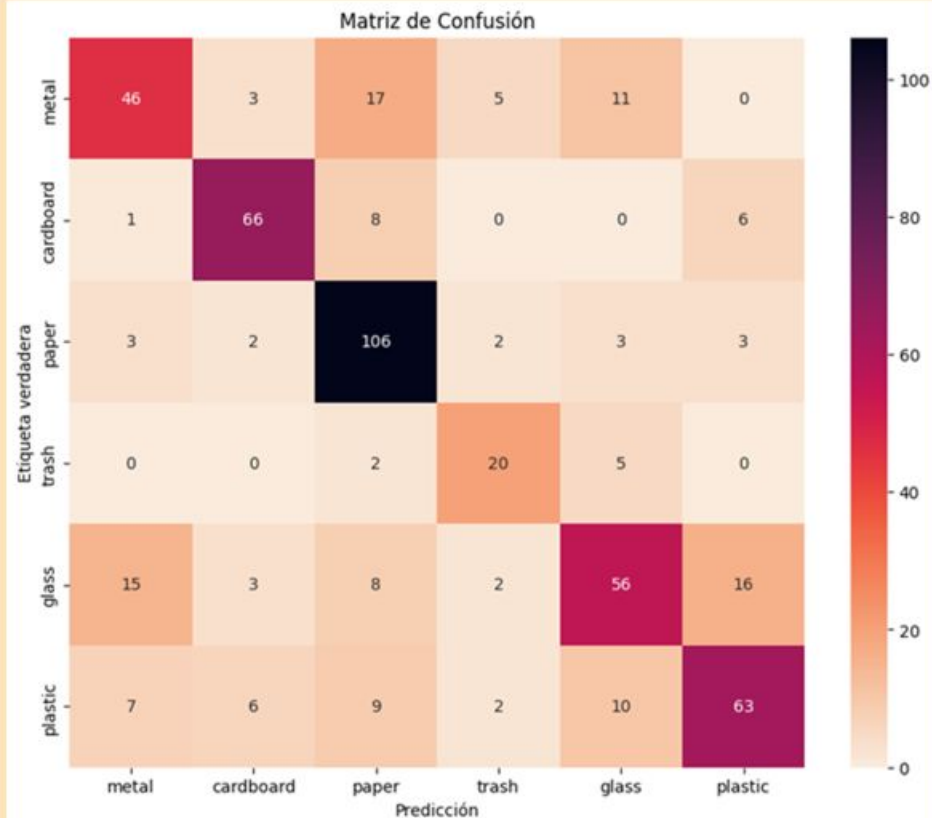
## Modelos Entrenados:

- **CNN sin data augmentation:** desempeño modesto, con signos evidentes de overfitting.
- **CNN con data augmentation:** mejora en la capacidad de generalización, aunque con cierta inestabilidad durante el entrenamiento.
- **ResNet50 con fine-tuning parcial:** buenos resultados en pocas épocas
- **ResNet50 con fine-tuning completo:** se obtuvo el mejor rendimiento global, con mejoras consistentes en las métricas.

## Métricas de Evaluación:

- **Accuracy**
- **Precisión, Recall y F1-score**, calculados por clase.
- **Matriz de confusión**, para análisis detallado de errores.
- **Curvas ROC y valores de AUC**, para evaluar el comportamiento del modelo.

# CNN sin data augmentation



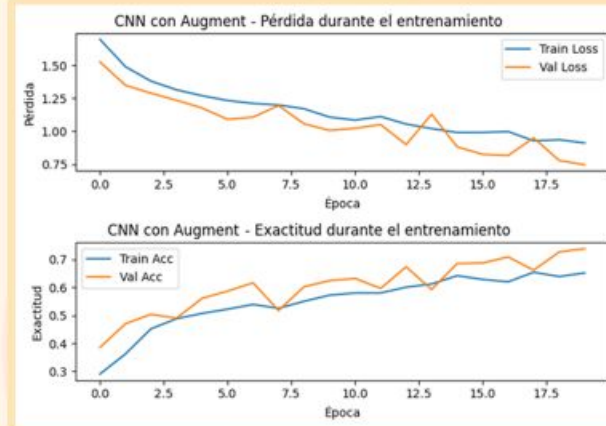
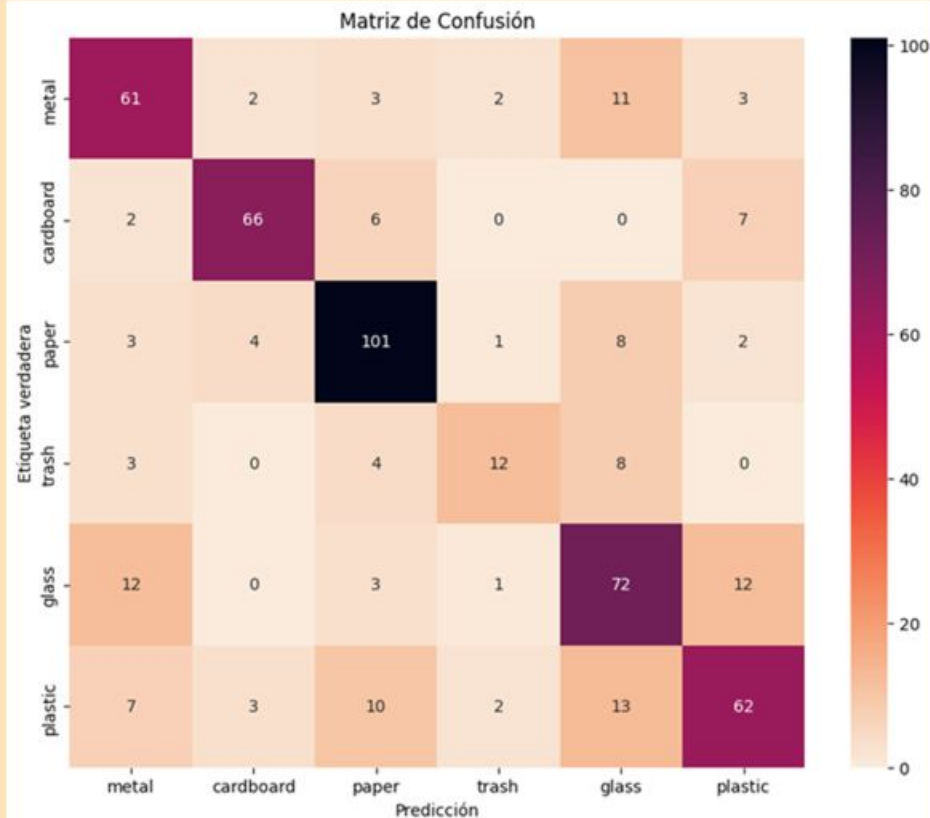
Reporte de Clasificación:

	precision	recall	f1-score	support
metal	0.64	0.56	0.60	82
cardboard	0.82	0.81	0.82	81
paper	0.71	0.89	0.79	119
trash	0.65	0.74	0.69	27
glass	0.66	0.56	0.61	100
plastic	0.72	0.65	0.68	97
accuracy			0.71	506
macro avg	0.70	0.70	0.70	506
weighted avg	0.70	0.71	0.70	506

# CNN sin data augmentation

- El desempeño fue **modesto**, con **métricas aceptables**, aunque se observó **confusión** en la **clasificación** de **ciertos tipos** de **residuos**.
- Se evidenció **overfitting**, dado que la **precisión** en **entrenamiento** fue considerablemente mayor que en **validación**, y la **pérdida en validación** permaneció **elevada**.
- El modelo **no logró una buena generalización**, probablemente debido a la **falta de variabilidad** en los **datos de entrenamiento**, lo cual **refuerza la necesidad** de **aplicar técnicas de data augmentation**.

# CNN con data augmentation



	precision	recall	f1-score	support
metal	0.69	0.74	0.72	82
cardboard	0.88	0.81	0.85	81
paper	0.80	0.85	0.82	119
trash	0.67	0.44	0.53	27
glass	0.64	0.72	0.68	100
plastic	0.72	0.64	0.68	97
accuracy			0.74	506
macro avg	0.73	0.70	0.71	506
weighted avg	0.74	0.74	0.74	506

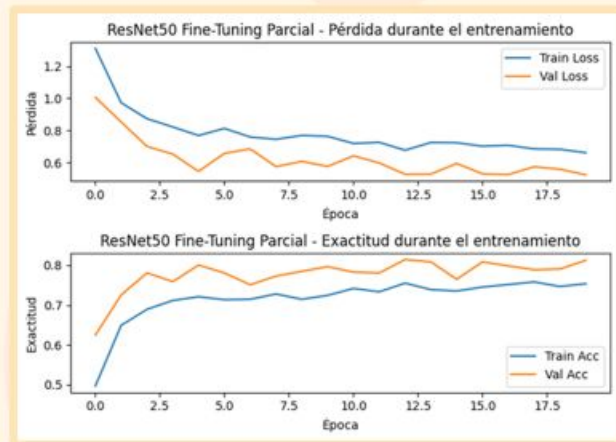
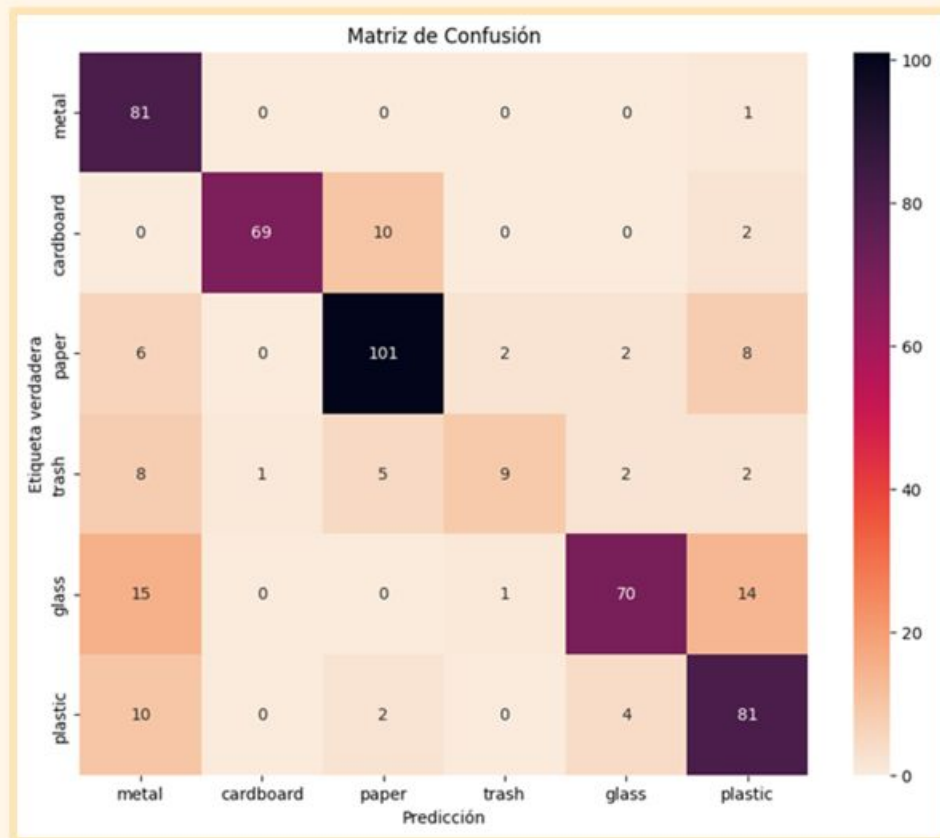


# CNN con data augmentation

## Impacto del Data Augmentation:

- La incorporación de **data augmentation** permitió **reducir el overfitting**, al introducir mayor variabilidad en los datos de entrenamiento.
- Si bien la **precisión en entrenamiento disminuyó levemente**, el modelo mostró una **mejor capacidad de generalización**, con un desempeño más equilibrado entre los conjuntos de entrenamiento y validación.

# ResNet50 con fine-tuning parcial



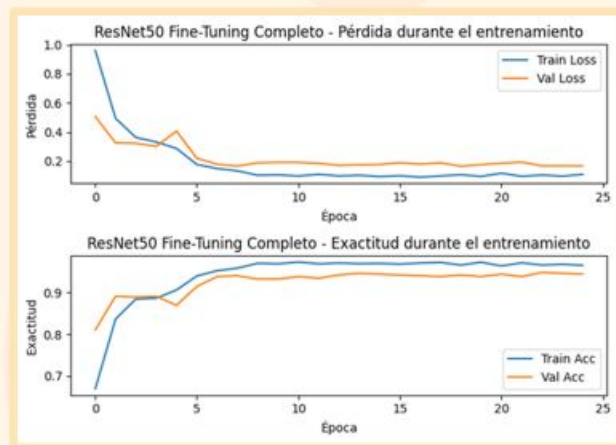
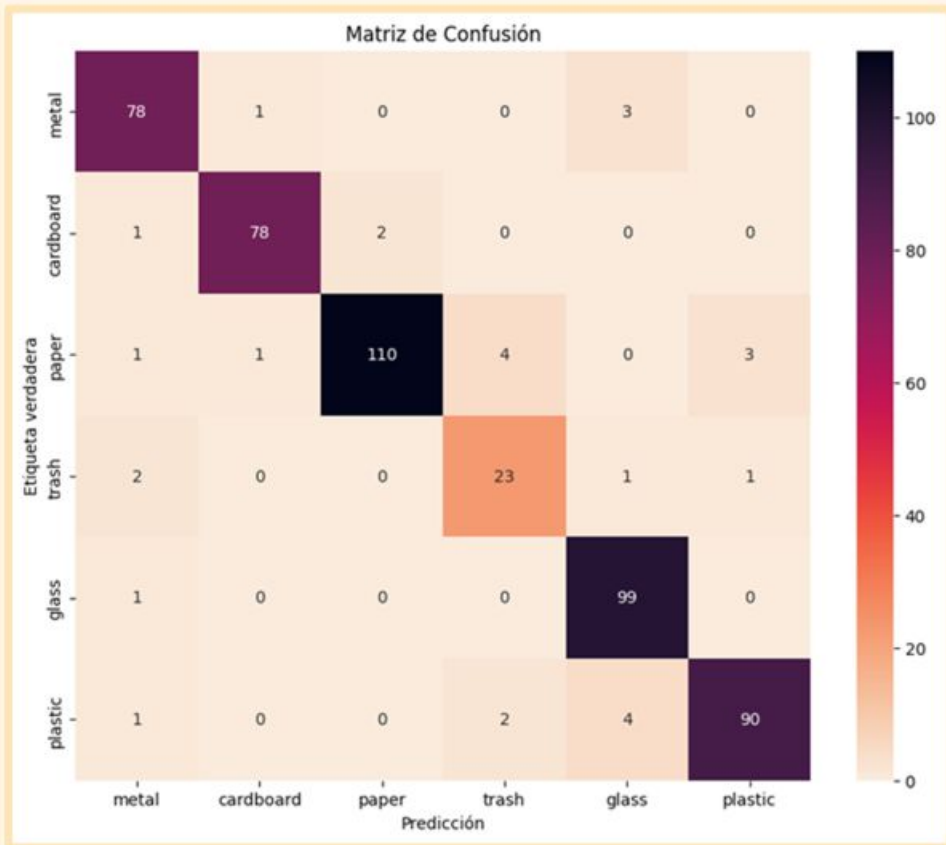
Reporte de Clasificación:

	precision	recall	f1-score	support
metal	0.68	0.99	0.80	82
cardboard	0.99	0.85	0.91	81
paper	0.86	0.85	0.85	119
trash	0.75	0.33	0.46	27
glass	0.90	0.70	0.79	100
plastic	0.75	0.84	0.79	97
accuracy			0.81	506
macro avg	0.82	0.76	0.77	506
weighted avg	0.83	0.81	0.81	506

# ResNet50 con fine-tuning parcial

- El modelo obtuvo un **buen rendimiento general**, especialmente destacable considerando que se entrenó con **solo 20 epochs**.
- Al utilizar **pesos preentrenados (IMAGENET1K\_V1)**, el modelo parte de una **base sólida de representación de características**, lo que permite una **rápida mejora en el aprendizaje**.
- Presenta un **excelente equilibrio entre costo computacional y desempeño**, lo cual lo convierte en una opción **eficiente para entornos con recursos limitados**.

# ResNet50 con fine-tuning completo



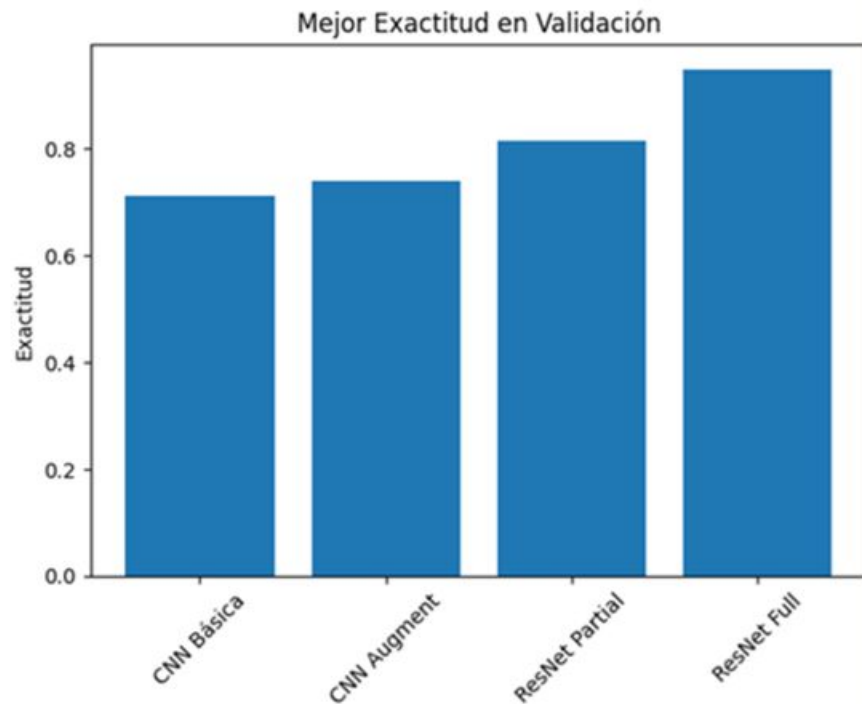
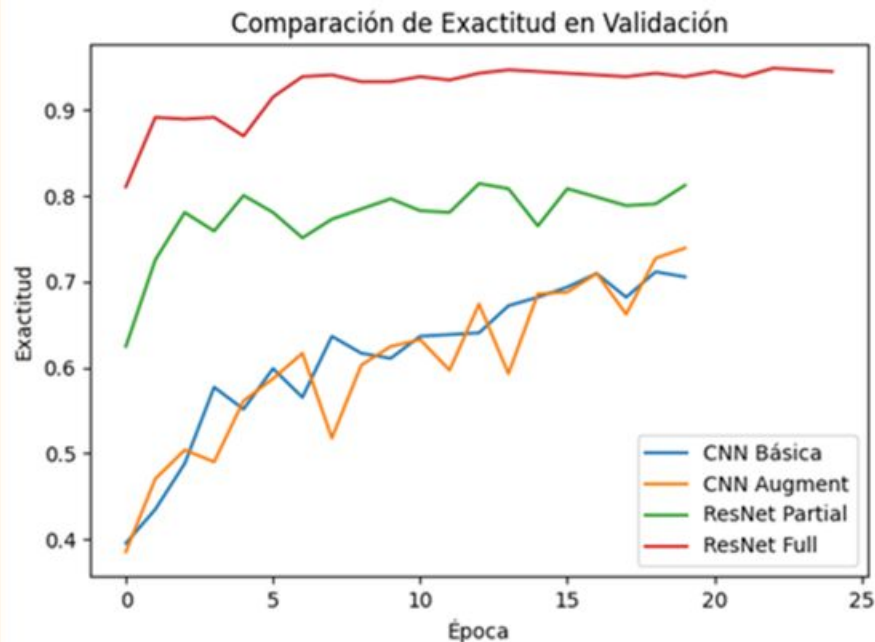
Reporte de Clasificación:

	precision	recall	f1-score	support
metal	0.93	0.95	0.94	82
cardboard	0.97	0.96	0.97	81
paper	0.98	0.92	0.95	119
trash	0.79	0.85	0.82	27
glass	0.93	0.99	0.96	100
plastic	0.96	0.93	0.94	97
accuracy			0.94	506
macro avg	0.93	0.93	0.93	506
weighted avg	0.95	0.94	0.94	506

# ResNet50 con fine-tuning completo

- Es el modelo con mejor desempeño entre todos los evaluados.
- Se observó una alta precisión en validación, pérdida baja y curvas de aprendizaje estables, lo que indica un entrenamiento sólido.
- El ajuste completo de los pesos permitió una adaptación total al dominio del problema, maximizando la capacidad predictiva del modelo.

# Discusión y Análisis



# Discusión y Análisis

## Comparación de Modelos – Desempeño General:

- **CNN Básica:** muestra una mejora progresiva pero limitada, alcanzando un techo de exactitud cercano al 65%.
- **CNN con Data Augmentation:** logra una ligera mejora respecto a la CNN básica, evidenciando mejor generalización pero sin un salto significativo en el rendimiento.
- **ResNet50 con Fine-Tuning Parcial:** parte de una buena precisión inicial y mejora con rapidez, aunque presenta cierta inestabilidad durante el entrenamiento.
- **ResNet50 con Fine-Tuning Completo:** domina desde el inicio y logra una exactitud superior al 90%, con un entrenamiento estable y consistente, mostrando la mejor capacidad de generalización entre todos los modelos evaluados.

# Optimización y ajuste de hiperparámetros

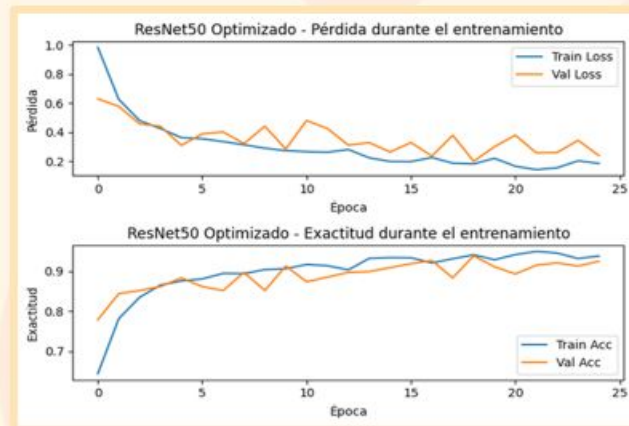
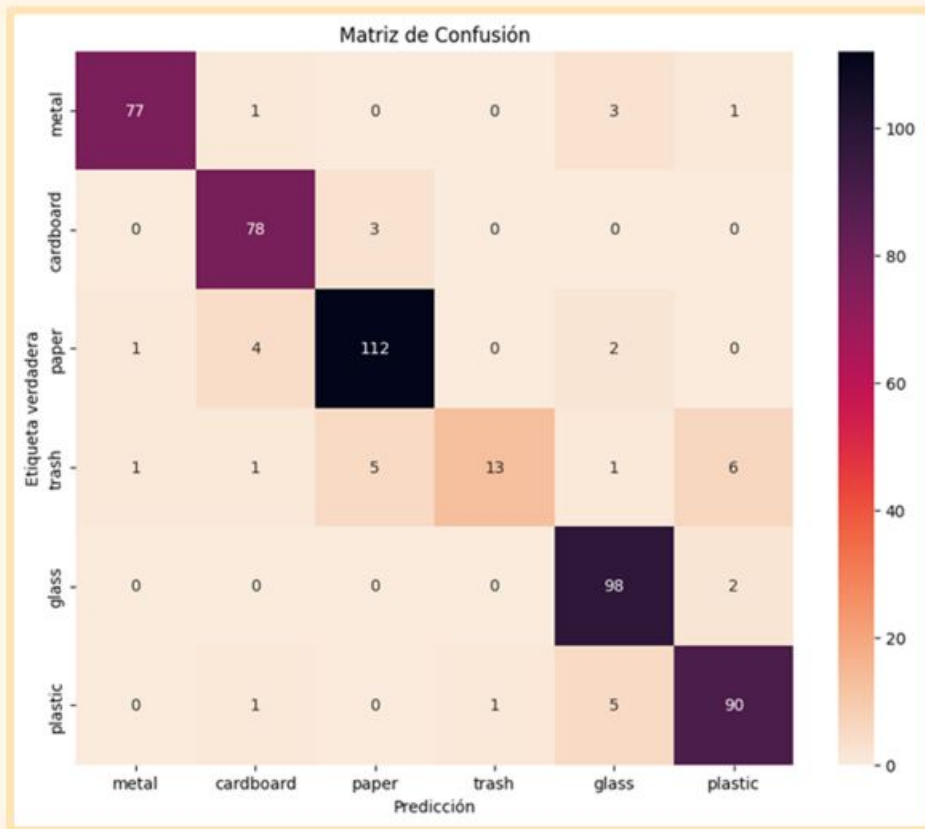
Para el ajuste de hiperparámetros se hizo uso de Optuna (búsqueda bayesiana) en 10 trials.

- **Búsqueda más inteligente** que random/grid search.
- **Más rápido**: menos modelos entrenados, pero más efectivos.
- **Podado temprano (pruning)**: evita perder tiempo en combinaciones malas.

Hiperparámetro	Tipo de Búsqueda	Rango / Valores	Descripción	Valores óptimos
lr	Float (log)	1e-5 – 1e-3	Tasa de aprendizaje.	0.0005902219411187059
optimizer	Categorical	'Adam', 'SGD'	Tipo de optimizador.	Adam
Weight decay	Float (log)	1e-6 – 1e-2	Regularización L2.	0.0001252367872397
dropout rate	Float	0.1 – 0.5	Dropout en la cabeza del modelo.	0.28355204189043265
Batch size	Categorical	16, 32, 64	Tamaño de mini-batch.	32
scheduler	Categorical	'ReduceLROnPlateau', 'CosineAnnealing', None	Tipo de scheduler.	CosineAnnealing



# ResNet50 con Optuna (20 trials)



## Reporte de Clasificación:

	precision	recall	f1-score	support
metal	0.97	0.94	0.96	82
cardboard	0.92	0.96	0.94	81
paper	0.93	0.94	0.94	119
trash	0.93	0.48	0.63	27
glass	0.90	0.98	0.94	100
plastic	0.91	0.93	0.92	97
accuracy			0.92	506
macro avg	0.93	0.87	0.89	506
weighted avg	0.93	0.92	0.92	506

# Trabajo futuro

## Próximos Pasos:

- Continuar con el **ajuste de hiperparámetros** utilizando **Optuna**, enfocándose en **otras arquitecturas**, como por ejemplo, la **tasa de aprendizaje (lr)** en la **CNN** **con** **data** **augmentation**.
- **Explorar nuevos datasets** y probar **arquitecturas más avanzadas**, con el fin de mejorar la precisión y generalización del modelo.

# Gracias !



**CREDITS:** This presentation template was created by [Slidesgo](#), and includes icons by [Flaticon](#), and infographics & images by [Freepik](#)

Please keep this slide for attribution

