

Assessing advanced computer vision for waste image classification

Juan Manuel Camara Diaz

Resumen— El reciclaje es esencial para proteger el medio ambiente y conservar los recursos naturales al transformar materiales utilizados en nuevos productos y reducir la acumulación de desechos en vertederos. La automatización del proceso de reciclaje, tanto a nivel doméstico como profesional, puede incrementar la eficiencia y exactitud en la separación de residuos. Este artículo examina avances en visión por computadora aplicados al desarrollo de herramientas para la clasificación de desechos. Se aborda la creación de instrumentos para simplificar el etiquetado de imágenes de residuos, así como el desarrollo de una aplicación móvil destinada a promover y facilitar el reciclaje en el hogar. Por último, se evalúa la factibilidad de construir un conjunto de datos de segmentación utilizando imágenes etiquetadas.

Palabras clave— Reciclaje, automatización, visión por computación, redes neuronales, generación de imagen, lenguaje natural, clasificación, segmentación, dataset, aplicación.

Abstract— Recycling is essential for protecting the environment and conserving natural resources by transforming used materials into new products and reducing the accumulation of waste in landfills. The automation of the recycling process, both at the household and professional level, can increase efficiency and accuracy in waste separation. This article examines advances in computer vision applied to the development of tools for waste classification. The creation of instruments to simplify labeling of waste images is addressed, as well as the development of a mobile application aimed at promoting and facilitating recycling at home. Finally, the feasibility of constructing a segmentation dataset using labeled images is evaluated.

Keywords— Recycling, automation, computer vision, neural networks, image generation, natural language, classification, segmentation, dataset, application.

1 INTRODUCCIÓN

El reciclaje es el proceso de convertir materiales usados en nuevos productos para reducir la cantidad de residuos que van a parar a los vertederos y minimizar el impacto ambiental. El reciclaje es fundamental para proteger el medio ambiente y preservar los recursos naturales.

En primer lugar, el reciclaje ayuda a reducir la cantidad de residuos que van a parar a los vertederos, lo que a su vez reduce la cantidad de espacio que se necesita para depositar-los. Los vertederos ocupan un espacio valioso y pueden ser una fuente de contaminación del aire y del agua si no se gestionan adecuadamente. Además, los residuos que se depositan en los vertederos pueden tardar décadas o incluso siglos en descomponerse, lo que aumenta aún más el problema.

En segundo lugar, el reciclaje ayuda a conservar los recursos

naturales. Muchos de los productos que utilizamos diariamente, como el papel, el vidrio y el plástico, están hechos de recursos naturales limitados, como los árboles, el petróleo y el gas natural. Al reciclar estos materiales, se reduce la necesidad de extraer nuevos recursos naturales, lo que a su vez reduce la huella de carbono y ayuda a preservar los recursos para las generaciones futuras.

Por otro lado, la automatización del reciclaje, tanto a nivel doméstico como profesional, puede mejorar significativamente la eficiencia y la precisión del proceso de separación de residuos. En el caso de la automatización doméstica, existen herramientas y dispositivos que pueden ayudar a clasificar los productos y saber en qué contenedor deben ir. Por ejemplo, contenedores con sensores de luz y peso que indican al usuario si un objeto debe ir al contenedor de papel, vidrio o plástico.

En cuanto a la automatización profesional, las cintas transportadoras y otras herramientas mecánicas pueden separar los residuos en diferentes categorías de manera más rápida y precisa que si se hiciera de forma manual. Además, la automatización puede reducir la cantidad de mano de obra necesaria para separar los residuos, lo que a su vez reduce los costos y mejora la eficiencia del proceso.

Por estos motivos me he enfocado en explorar los nuevos avances

- E-mail de contacto: juanma.caaz@gmail.com
- Mención realizada: Ingeniería de Computación
- Trabajo tutorizado por: Coen Antens (CVC)
- Curso 2022/2023

que han ido saliendo en el campo de la visión por computación para aportar herramientas para la clasificación de residuos. En los últimos años, se han logrado importantes avances en el campo de la visión por computador gracias a los modelos de redes neuronales que se pueden utilizar para resolver problemas de clasificación relacionado con el reciclaje. [1]

2 MOTIVACIÓN

En 2020 se aprobó el Pacto Verde Europeo [2], también conocido como green deal y que tiene como objetivo lograr la neutralidad climática en Europa para el año 2050. Este plan ambicioso busca transformar la economía europea y abordar los desafíos del cambio climático y la pérdida de biodiversidad. Incluye medidas para mejorar la eficiencia energética, reducir las emisiones de gases de efecto invernadero y fomentar la transición hacia una economía circular. Además, el Pacto Verde Europeo se enfoca en la creación de empleos y en el apoyo a una transición justa para garantizar que todos los ciudadanos europeos se beneficien de una economía más sostenible. Por lo tanto creo que realizar una clasificación de estos residuos puede ayudar a cumplir con este objetivo.

La clasificación de residuos es un problema importante en la gestión de residuos y en la preservación del medio ambiente. Sin embargo, la eficiencia de los modelos de clasificación existentes todavía puede mejorarse. En este trabajo de investigación, se abordarán diversas cuestiones relacionadas con la clasificación de residuos utilizando técnicas de inteligencia artificial.

Para aportar soluciones a este problema, se han definido los siguientes objetivos:

1. Generar un gran dataset para cubrir la mayor cantidad de residuos reciclables y no reciclables utilizando utilizando una aplicación móvil para etiquetar de manera mas colaborativa.
2. Facilitar el reciclaje de residuos utilizando una aplicación movil que permita a los usuarios clasificar sus residuos de manera automática superando las aplicaciones domésticas actuales.
3. Explorar la viabilidad de generar un dataset semántico a partir de uno de clasificación, para reducir el tiempo y coste del etiquetado.

Con estos objetivos se busca mejorar la clasificación de residuos y contribuir a la gestión sostenible de los mismos.

3 ESTADO DEL ARTE

En los últimos años, se han logrado importantes avances en el campo del Deep Learning, esto gracias a la aparición de los Transformers. Presentados en el paper de Attention is all you need [4].

Los transformers son clave en modelos de inteligencia artificial porque han demostrado ser muy efectivos en tareas de procesamiento del lenguaje natural (NLP, por sus siglas en inglés) y otras aplicaciones relacionadas con el aprendizaje automático, incluso extrapolando el su uso a todo tipo de tareas.

3.1. Clasificación

La clasificación ha dado un cambio radical con la salida del CLIP [8]. CLIP es un modelo de aprendizaje profundo que ha cambiado el estado del arte en la clasificación de imágenes. Utiliza un modelo de lenguaje para entender tanto las imágenes como el texto relacionado y una técnica de aprendizaje por similitud para clasificar imágenes en función de su similitud con una determinada descripción textual.

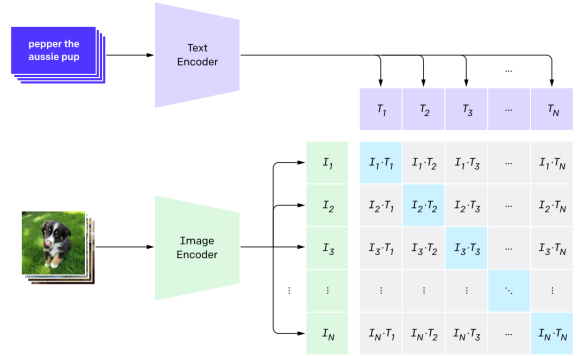


Fig. 1: Arquitectura de CLIP, creada por OpenAI.

En comparación con los modelos clásicos de redes neuronales convolucionales, CLIP puede comprender el contexto y el significado detrás de una imagen, lo que lo hace especialmente útil en situaciones en las que las imágenes pueden ser difíciles de clasificar para los modelos clásicos.

Modelos	Accuracy	Params	Technology
BASIC-L	91.1 %	2440M	Conv+Transf
CoCa	91.0 %	2100M	Transformer
Model soups	90.98 %	2440M	Conv+Transf
Model soups	90.94 %	1843M	Transformer
ViT-e	90.9 %	3900M	Transformer
CoAtNet-7	90.88 %	2440M	Conv+Transf
ViT-G/14	90.71 %	1843M	Transformer
CoCa	90.60 %	2100M	Transformer
CoAtNet-6	90.45 %	1470M	Conv+Transf
ViT-G/14	90.45 %	1843M	Transformer
DaViT-G	90.4 %	1437M	Transformer
Meta Pseudo Labels	90.2 %	480M	EfficientNet
DaViT-H	90.2 %	362M	Transformer
SwinV2-G	90.17 %	3000M	Transformer
Florence-CoSwin-H	90.05 %	893M	Transformer
Meta Pseudo Labels	90 %	390M	EfficientNet
RevCol-H	90.0 %	2158M	Pure CNN

Tabla 1: Comparativa de modelos de clasificación con mejor accuracy en imágenes del dataset Imagenet tabla extraída y adaptada de "paperswithcode". Modelos del 2020 al 2023-Q1.

En la tabla se puede ver una comparativa de los modelos de clasificación de imágenes más destacados del 2020 al 2023-Q1 y como los modelos basados en transformers (entrenados con el contexto) son superiores a los son puramente convolucionales. Esto se debe a que los modelos de redes neuronales convolucionales puros no pueden comprender el contexto y el significado detrás de una imagen, mientras que los modelos de redes neuronales basados en Transformer pueden comprender el contexto y el significado detrás de una imagen consiguiendo mejores resultados.

También es interesante ver que modelos con una cantidad de parámetros muy inferior a los modelos de redes neuronales convolucionales puros, como el modelo de ViT-G/14, pueden conseguir resultados mejores que los modelos de redes neuronales convolucionales puros.

Respecto a la clasificación del lenguaje natural, también se ha visto que los transformers son las arquitecturas que mejor funcionan dejando atrás a los modelos LSTM.

3.2. Visualización de Redes Neuronales

La visualización de las redes neuronales se ha vuelto esencial para comprender los modelos de aprendizaje profundo. Existen diversas técnicas notables en este campo.

3.2.1. Visualización de Características

Activation Maximization (AM) es un procedimiento que se utiliza para explorar y entender los modelos de redes neuronales profundas, a través de la identificación de estímulos de entrada que maximizan la activación de ciertas neuronas en el modelo. En el contexto de las redes neuronales, una 'activación' se refiere a la salida de una neurona después de procesar una entrada dada. Cada neurona en la red está sintonizada para reaccionar a ciertos patrones en los datos de entrada. Algunas neuronas pueden responder a patrones de bajo nivel, como bordes o colores en una imagen, mientras que otras pueden responder a patrones más complejos y abstractos. El propósito de la Activación Máxima es descubrir cuál es este patrón para una neurona específica. Para hacer esto, AM ajusta iterativamente una entrada hasta que encuentra una que maximiza la activación de la neurona objetivo. Esta entrada es entonces una representación de los patrones que la neurona está diseñada para detectar.

3.2.2. Mapas de Activación de Clase

El mapa de activación de clase (CAM) es una técnica de visualización de características que permite visualizar qué partes de una imagen son importantes para una clase en particular. CAM se puede utilizar para cualquier red neuronal convolucional (CNN) con una capa de agrupación global (GAP) en la parte superior.

- **GradCAM:** Utiliza los gradientes de cualquier capa objetivo para producir un mapa de activación de clase. GradCAM utiliza los gradientes de la clase de salida con respecto a las características de la capa para ponderar la importancia de cada característica para la decisión de la clase.
- **GradCAM++:** Una mejora de GradCAM que captura más detalles y es más preciso. GradCAM++ modifica la ponderación de los gradientes para capturar patrones más detallados y diferentes tipos de características, incluyendo características de alto nivel y de bajo nivel.
- **ScoreCAM:** Se basa en una serie de modificaciones del enfoque original de GradCAM para evitar problemas de co-localización. En lugar de utilizar gradientes, ScoreCAM ajusta cada canal de características de la capa objetivo y pasa las

imágenes ajustadas a través de la red para obtener puntuaciones.

- **Faster-ScoreCAM:** Una versión acelerada de ScoreCAM que reduce significativamente el tiempo de computacional al evitar la necesidad de pasar las imágenes ajustadas por toda la red.
- **LayerCAM:** Combina la evidencia de múltiples capas de la red para generar un mapa de activación de clase más preciso. LayerCAM introduce un factor de ponderación que da más importancia a las características de alto nivel en comparación con las características de bajo nivel.

3.2.3. Mapas de Saliencia

Los mapas de saliencia son una técnica de visualización de características que permite visualizar qué partes de una imagen son importantes para una clase en particular. Los mapas de saliencia se pueden utilizar para cualquier (CNN).

- **Vanilla Saliency:** Para calcular un mapa de saliencia, lo que se hace es calcular la derivada (o gradiente) de la salida de la red con respecto a la entrada. Este cálculo indica cuánto cambiaría la salida de la red si se cambia ligeramente un píxel en la imagen de entrada. Si este cambio es grande, entonces ese píxel es importante para la clasificación de la red, y se decolorará fuertemente en el mapa de saliencia. Si este cambio es pequeño, entonces ese píxel no es muy importante para la clasificación de la red, y será menos colorido en el mapa de saliencia. Este tipo de técnica suele generar ruido en la salida, es decir puede haber píxeles que no sean importantes pero que se colorean fuertemente generando inconsistencias.
- **SmoothGrad:** Es una técnica que se utiliza para mejorar la calidad y la interpretación de los mapas de saliencia generados por las redes neuronales. Los mapas de saliencia, como mencioné antes, pueden ser ruidosos o inconsistentes debido a las complejidades de las redes neuronales. Para superar este problema, SmoothGrad introduce ruido gaussiano a las entradas de la red. El ruido gaussiano es esencialmente una perturbación aleatoria con cierta variabilidad, que se añade a cada píxel de la imagen de entrada. Este proceso de añadir ruido se repite muchas veces, y para cada versión con ruido de la imagen, se calcula un mapa de saliencia. Luego, estos mapas se promedian para obtener un mapa de saliencia "suavizado". El resultado de este proceso es un mapa de saliencia más suave y menos ruidoso, que puede ser más fácil de interpretar y entender. Esto es útil para analizar cuáles partes de la imagen son más importantes para la decisión final de la red neuronal.

A medida que los modelos de aprendizaje profundo se vuelven cada vez más complejos, las técnicas de visualización y los algoritmos se vuelven cada vez más necesarios para interpretar estos modelos.

Por lo general los mapas de Saliencia suele ser mas precisos que los mapas de activación de clase, pero son mas costosos computacionalmente.

3.3. Segmentación Semántica

La segmentación semántica es una tarea de clasificación de píxeles que tiene como objetivo asignar a cada píxel en una imagen una etiqueta de clase específica. Formalmente, dada una imagen de entrada I de tamaño $W \times H \times C$ (donde W es el ancho, H es la altura y C son los canales de color), el objetivo es generar un mapa de segmentación S de tamaño $W \times H$ donde cada ubicación s_{ij} en S contiene la etiqueta de clase de la ubicación correspondiente en la imagen de entrada.

Los métodos convencionales para la segmentación semántica incluyen técnicas como el crecimiento de regiones y la segmentación por aguas divisorias. Sin embargo, con el surgimiento de las redes neuronales convolucionales (CNNs), los enfoques basados en el aprendizaje profundo se han convertido en el estándar de oro para la segmentación semántica.

Uno de los modelos más influyentes en este campo es la Red SegNet, que es una arquitectura de red neuronal convolucional completamente convolucional para la segmentación semántica. La arquitectura de SegNet se puede describir como sigue: dada una imagen de entrada I , se aplica una serie de capas convolucionales para producir una representación de características F de tamaño reducido. Luego, se aplica un conjunto de capas de decodificación a F para producir el mapa de segmentación de salida S .

La función de pérdida que se utiliza comúnmente para entrenar estos modelos es la entropía cruzada, que se puede formular como sigue:

$$L(\theta) = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^J y_{ij} \log(\hat{y}_{ij}), \quad (1)$$

donde N es el número de píxeles en la imagen, J es el número de clases, y_{ij} es la etiqueta de clase real para el píxel i en la clase j , y \hat{y}_{ij} es la probabilidad predicha por el modelo para el píxel i en la clase j .

Los modelos modernos de segmentación semántica, como U-Net, Mask R-CNN y DeepLab, han extendido y mejorado estas ideas básicas, proporcionando un rendimiento de segmentación semántica de vanguardia en una variedad de tareas y dominios.

Cabe mencionar que, aunque la segmentación semántica ha logrado avances significativos, todavía se enfrenta a desafíos. Por ejemplo, los objetos pequeños y las regiones de bordes finos a menudo son difíciles de segmentar correctamente. Además, la necesidad de anotaciones de píxeles completas para el entrenamiento puede ser costosa en términos de tiempo y recursos humanos. En la siguiente sección, discutiremos cómo la segmentación débilmente supervisada (WSSS) intenta abordar este último desafío.

3.4. Weakly Supervised Semantic Segmentation (WSSS)

La segmentación débilmente supervisada (WSSS) es una técnica que nos ayuda a entender qué partes de una imagen pertenecen a qué objeto o categoría, sin necesidad de que alguien nos diga exactamente qué píxel pertenece a qué objeto, lo que normalmente requeriría mucho trabajo y tiempo.

Imaginamos que tenemos una imagen de un perro jugando en un parque. La etiqueta de la imagen podría ser simplemente “perro”,

en lugar de tener un etiquetado para cada píxel indicando si es parte del perro, del césped, del árbol, etc.

¿Cómo hace la WSSS para entender esto? Primero, se usa una red neuronal convolucional (CNN) para extraer un “mapa” de todas las características que encuentra.

Después, para cada píxel en la imagen, revisa este mapa y, basado en las características que encuentra, decide si ese píxel es parte del “perro” o no. Si la característica relacionada con ese píxel es muy fuerte, tiene alta “activación”, entonces se decide que ese píxel pertenece al “perro”.

Aunque esta técnica suena muy útil, tiene algunos problemas. Por ejemplo, la calidad de la segmentación depende mucho de lo buena que sea la CNN para identificar las características correctas. A veces puede confundirse y no funciona igual de bien en todos los tipos de imágenes o para todas las categorías de objetos.

Por este último motivo se va a explorar a utilizar Segment Anything con el objetivo de mejorar la segmentación de las imágenes.

4 METODOLOGÍA

Para la realización del proyecto se han determinado unas metodologías de trabajo, desde a nivel de planificación hasta a nivel de desarrollo.

4.1. Planificación

Para la planificación he decidido realizar tareas que sean ejecutadas secuencialmente para así poder llevar un mayor control del avance del proyecto. Cada fase esta compuesta de diferentes tareas a realizar. Se puede ver en el Apéndice la planificación detallada.

La metodología a utilizar en el proyecto sera Kanban, cada tarea se representara en una tarjeta Kanban y se moverá a la columna correspondiente en función de su estado. Los estados que se han definido son:

TODO: tarea pendiente de realizar.
IN PROGRESS: tarea en proceso.
DONE: tarea realizada.

Para la implementación de la metodología Kanban se ha utilizado la herramienta Trello, que permite crear tableros y gestionar las tareas de manera sencilla. En el Apéndice se puede ver el tablero Kanban.

Para todo el código desarrollado se utilizara el sistema de control de versiones Git para llevar un control de las versiones del proyecto.

4.2. Tecnologías

Para llevar a cabo el proyecto, se han seleccionado una serie de tecnologías que permitirán realizar de manera eficiente el procesamiento de imágenes y el análisis de datos, así como el despliegue de modelos de aprendizaje automático.

- OpenCV: una biblioteca de visión por computadora de código abierto que proporciona herramientas para el procesamiento de imágenes y el análisis de vídeo en tiempo real. Esta

tecnología se eligió debido a su capacidad para proporcionar una amplia gama de herramientas para el procesamiento de imágenes y su eficiente despliegue de modelos de aprendizaje profundo (DNN).

- Diversas bibliotecas de Python, incluyendo SKlearn, numpy, matplotlib y pandas, para el tratamiento de datos y visualizaciones. Estas bibliotecas ofrecen un conjunto de herramientas y funcionalidades para la manipulación de datos y su representación gráfica.
- Firebase: para el despliegue de la aplicación móvil. Firebase es una plataforma de desarrollo de aplicaciones móviles y web que ofrece una amplia gama de herramientas para el desarrollo de aplicaciones.
- Meta's Segment Anything Model [22]: para el procesamiento de imágenes. Este modelo de aprendizaje profundo permite segmentar objetos en imágenes dando como entrada una imagen y puntos de referencias.
- Tensorflow2 [10]: para el entrenamiento y creación de modelos multimodales. Tensorflow2 es una biblioteca de aprendizaje automático de código abierto que se utiliza ampliamente en la creación de modelos de aprendizaje profundo y su despliegue en producción.
- OpenCLIP: Una versión open source de CLIP [8], que es un modelo de aprendizaje profundo que permite clasificar imágenes.
- Colab [11]: como entorno de ejecución durante la mayor parte del proyecto. La versión pro de Colab permite un mayor tiempo de ejecución y recursos computacionales, lo que es necesario para el procesamiento de imágenes y la creación de modelos de aprendizaje profundo.
- Android Studio: para el desarrollo de la aplicación móvil. Esta plataforma de desarrollo proporciona herramientas para la creación de aplicaciones para dispositivos móviles y su despliegue en dispositivos Android.
- AIRE: Chatbot creado por ecoenbes, es la única aplicación que permite usar imágenes para clasificar residuos a nivel doméstico.

4.3. Documentación

La documentación se realizó en el lenguaje de marcado LaTeX, ya que es un lenguaje de marcado que permite crear documentos de gran calidad y que es muy utilizado en la comunidad científica.

Parte de la documentación se encontrará en el repositorio del proyecto en GitHub y todo el dataset generado será publicado y de libre acceso en el repositorio.

5 DESARROLLO

5.1. Dataset

Unos de los principales problemas para la generación del dataset es que en cada región tiene un sistema de reciclaje diferente y por lo tanto las clases de residuos que se pueden reciclar también son diferentes por lo que ha sido necesario crear un dataset propio adaptado a Cataluña. El motivo de utilizar el sistema de Cataluña

es porque llevan tiempo fomentando la separación de residuos y hay mucha información disponible sobre el tema.

Residuonvas.cat es una página web que tiene como objetivo informar a la población sobre la gestión de residuos en Cataluña. En su sitio web, promueven un sistema de clasificación y separación de residuos en diferentes categorías para su posterior tratamiento y reciclaje. A continuación, se presenta una lista de las diferentes clases de residuos para reciclar que se promueven en este sistema:

Envase de vidrio	Envase ligero
Medicamentos	Pilas y baterías
Aparatos eléctricos	Ropa y calzado
Punto verde	Orgánica
	Resto

Esta lista serán las clases que se utilizarán para la creación del dataset.

Para la recolección del dataset se ha utilizado diferentes dataset de residuos para tener una base sólida para poder entrenar el modelo. También se ha desarrollado una aplicación móvil para la recolección de imágenes de residuos para poder tener un dataset más completo y concreto de residuos domésticos. En la tabla 2 se puede ver la distribución de las clases en el dataset.

Clase	Train	%	Val	%	Total
Amarillo	7371	15.15	500	10.64	7871
Azul	4230	8.7	500	10.64	4730
Marrón	4980	10.24	500	10.64	5480
Medicamento	3156	6.49	400	8.51	3556
Pilas	3214	6.61	300	6.38	3514
Punto	4576	9.41	500	10.64	5076
RAEE	6258	12.87	500	10.64	6758
Resto	4443	9.13	500	10.64	4943
Ropa	5332	10.96	500	10.64	5832
Verde	5081	10.45	500	10.64	5581
Total	48641	100	4700	100	53341

Tabla 2: Distribución de las clases y en los conjuntos de entrenamiento y validación

El motivo de no tener de test es porque se está esperando a que los usuarios de la aplicación móvil contribuyan con más imágenes para poder tener un dataset más completo y concreto de residuos domésticos.

Para la búsqueda del mejor modelo de momento se utilizará el conjunto de validación. Una vez tengamos el mejor modelo se utilizará el conjunto de test, que está formado por imágenes que no han sido utilizadas en ningún momento para el entrenamiento ni validación del modelo y que son imágenes reales de un uso doméstico.

En la figura 2 se puede ver una imagen de cada clase extraída del conjunto de entrenamiento.

5.2. Aplicación móvil

Para poder recolectar imágenes domésticas reales de residuos se ha desarrollado una aplicación móvil para Android llamada EcoMate.



Fig. 2: Imágenes de cada clase extraídas del conjunto de entrenamiento.

Ha sido diseñada para permitir a los usuarios contribuir y mantener un dataset de imágenes en tiempo real de manera sencilla. Esta aplicación fomenta la colaboración y el aprendizaje en torno a la clasificación y manejo de residuos, y está disponible para su descarga en la Play Store.

Las principales funcionalidades de la aplicación son:

1. Listado en tiempo real: EcoMate muestra la cantidad de imágenes que los usuarios han contribuido al dataset, permitiendo a los usuarios mantenerse actualizados sobre el progreso de la comunidad.
2. Ranking de contribuyentes: La aplicación incluye un ranking que muestra la cantidad total de imágenes que ha contribuido el usuario en el primer puesto, fomentando la competencia amistosa y el compromiso con la plataforma.
3. Visualización de la última imagen: Los usuarios pueden ver la última imagen con la que han contribuido, lo que les permite revisar su progreso y mantenerse al tanto de sus aportaciones al dataset.
4. Carga de imágenes desde la cámara: EcoMate permite a los usuarios agregar imágenes nuevas directamente desde la cámara de su dispositivo móvil. Una vez tomada la foto, se asigna una clase del dataset y se carga en el servidor, actualizando automáticamente los datos en tiempo real.
5. Autenticación y seguridad: La aplicación utiliza Firebase como backend, con módulos de autenticación, Firestore Database, Storage y Realtime Database activos. Además, EcoMate ofrece inicio de sesión y registro con correo electrónico, verificación de correo electrónico y funcionalidad de “olvidé mi contraseña” para garantizar la seguridad y privacidad de los usuarios.

En el futuro, se planea agregar características adicionales, como la visualización de imágenes similares de residuos y la clasificación automática utilizando un modelo de clasificación avanzado. Estas mejoras enriquecerán aún más la experiencia del usuario y permitirán a la comunidad aprender y colaborar de manera más efectiva en torno al tema de la clasificación y gestión de residuos.

Desde que estuvo disponible en la Play Store, se ha recolectado un total de 398 imágenes de residuos.

Como se puede observar en la figura 3 la clase con más imágenes es la de AMARILLO, con 142 imágenes, mientras que la clase con menos imágenes es la de MEDICAMENTO. Esta información puede proporcionar una idea de cuáles son las clases más fáciles de obtener imágenes y cuáles son las más difíciles.

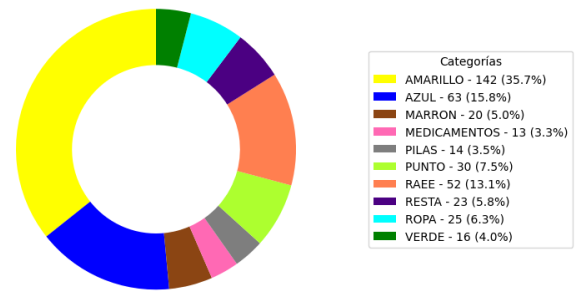


Fig. 3: Distribución de las clases en el dataset de la aplicación.

5.3. Modelo de clasificación

El principal objetivo de esta sección es identificar y seleccionar el modelo óptimo para llevar a cabo la clasificación de residuos de manera eficaz y precisa. Para lograrlo, se ha empleado la plataforma TensorFlow V2, en conjunto con la API de Keras, la cual facilita el diseño, entrenamiento y evaluación de modelos de aprendizaje profundo. Asimismo, se ha utilizado una tarjeta gráfica NVIDIA RTX 3060 con 12 GB de memoria de vídeo, permitiendo agilizar el proceso de entrenamiento y mejorar el rendimiento de los modelos.

En todas las pruebas realizadas se han usado los parámetros de la tabla 3.

Hiperparámetro	Valor
Optimizador	Adam
Learning rate	0.001
Amsgrad	True
Loss	Categorical crossentropy
Early stopping	Monitor: val_loss, Patience: 4
ModelCheckpoint	Monitor: val_loss, Save best only: True
Número de épocas	Hasta converger

Tabla 3: Hiperparámetros utilizados en las pruebas.

Se realizaron las primeras pruebas utilizando modelos simples, como una red neuronal multicapa (MLP) y una red neuronal convolucional (CNN). Estos modelos se entrenaron utilizando el conjunto de datos de entrenamiento sin realizar ninguna modificación en las imágenes.

Los valores analizados para evaluar el rendimiento de los modelos fueron la precisión con los datos de validación (Acc), la pérdida con los datos de validación (Loss), el número de parámetros (Params), el número de épocas (Epo) y el tiempo de ejecución por época en segundos (T/Epo).

Modelo	Acc	Loss	Params	Epo	Time/Epoch
simple_cnn	0.54	3.78	44.4M	7	184.61
simple_mlp	0.33	1.95	77.24M	22	71.35

Tabla 4: Resultados de las pruebas con modelos simples.

El modelo de red neuronal convolucional (CNN) ha demostrado un rendimiento superior como se observa en la tabla 4, ya que, al trabajar con imágenes, las CNN superan a las redes neuronales

multicapa (MLP). Además, se observa que el modelo CNN requiere un tiempo de entrenamiento por época significativamente mayor que el modelo MLP, a pesar de tener menos parámetros. Esto se debe a que el entrenamiento de capas convolucionales es mucho más costoso que el de capas densas. Sin embargo, el modelo CNN ha necesitado menos épocas para converger. Los resultados de esta primera prueba nos indican que es posible desarrollar un modelo de clasificación con una precisión aceptable si optamos por utilizar una red neuronal convolucional.

En la segunda prueba, se han utilizado modelos preentrenados con el conjunto de datos ImageNet. Para esta prueba inicial, se han congelado las capas de la red preentrenada y se han añadido una capa densa de 1024 neuronas y una capa de salida con 10 neuronas. De esta manera, podemos aprovechar el conocimiento previo de la red preentrenada y entrenar únicamente las capas añadidas.

Model	Acc	Loss	Params	Epo	T/Epo
xception[12]	0.83	0.69	123.63M	5	280.99
inceptionresnetv2[18]	0.83	0.78	93.67M	6	510.01
nasnetmobile[20]	0.81	0.73	57.27M	4	263.30
resnet50v2[13]	0.81	1.10	126.34M	4	291.10
mobilenetv2[16]	0.80	1.06	66.49M	4	114.87
vgg16[21]	0.74	1.36	40.42M	8	221.76
convnexttiny[19]	0.53	1.67	66.37M	6	727.44

Tabla 5: Resultados de las pruebas con modelos pre-entrenados y las capas convolucionales congeladas.

Si observamos la tabla ?? la mayoría de los modelos preentrenados han logrado una precisión superior al 80 %, lo que demuestra que el conocimiento adquirido a partir de las imágenes de ImageNet es de gran utilidad para clasificar las imágenes de nuestro conjunto de datos.

Cabe destacar que los modelos preentrenados requirieron menos épocas para converger en comparación con modelos anteriores. Esto se debe a que las capas convolucionales de los modelos preentrenados ya han aprendido a extraer características de las imágenes, por lo que solo necesitan aprender a clasificar las características extraídas.

Es importante mencionar que, a pesar de ser un modelo preentrenado, el modelo convnexttiny obtuvo una precisión muy baja en comparación con los demás.

En la tercera prueba, se llevó a cabo un entrenamiento completo de los modelos preentrenados, lo que implica descongelar las capas convolucionales y entrenar todo el modelo.

Para llevar a cabo este entrenamiento, se congelaron todas las capas convolucionales y se entrenaron únicamente la capa densa y la capa de salida durante cinco épocas. Posteriormente, se descongelaron todas las capas convolucionales y se entrenó todo el modelo hasta alcanzar la convergencia.

Además, se implementó el aumento de datos (data augmentation) debido a que, al descongelar todas las capas convolucionales, se incrementó el número de parámetros entrenables. Esto hace necesario aumentar el número de datos de entrenamiento para evitar el sobreajuste.

Las transformaciones aplicadas incluyen: rotación, traslación, volteo horizontal y ajuste aleatorio de brillo y contraste.

Se optó por utilizar un aumento de datos en tiempo real y diferente para cada época, aplicando transformaciones aleatorias a las imágenes de entrenamiento. Las probabilidades de aplicar cada transformación son:

Rotación: 0.5
 Traslación: 0.5
 Flip horizontal: 0.5
 Random brightness and contrast: 0.5

En la tabla 6 se muestran los resultados sin tener en cuenta el tiempo de entrenamiento de las 5 primeras épocas.

Modelo	Acc	Loss	Params	Epo	T/Epo
convnexttiny	0.91	0.36	66.37M	12	1799.02
xception	0.90	0.36	123.63M	10	670.20
inceptionresnetv2	0.90	0.38	93.67M	10	1149.01
nasnetmobile	0.87	0.54	57.27M	13	789.63
resnet50v2	0.86	0.54	126.34M	17	649.03
mobilenetv2	0.88	0.44	66.49M	15	358.09
vgg16	0.84	0.56	40.42M	22	689.81

Tabla 6: Resultados de las pruebas con modelos pre-entrenados y las capas convolucionales descongeladas.

Tres modelos de los siete mostrados han obtenido un accuracy superior al 90 % y lo mas importante es que el accuracy de convnexttiny ha mejorado considerablemente.

Finalmente se ha realizado una prueba con los modelos preentrenados sin data augmentation para comparar los resultados y demostrar que el data augmentation es necesario para obtener buenos resultados.

Modelo	Acc	Loss	Params	Epo	T/Epo
Xception con aug	0.90	0.36	123.63M	10	670.20
Xception sin aug	0.90	0.53	123.63M	6	808.19
MobileNetV2 con aug	0.88	0.44	66.49M	15	358.09
MobileNetV2 sin aug	0.88	0.70	66.49M	10	335.72

Tabla 7: Comparativa de modelos usando y sin usar data augmentation.

El data augmentation hace que el loss sea menor, esto quiere decir que el modelo es mas robusto y sera menos propenso al sobreajuste y generalizara mejor.

Sobre el tiempo de inferencia de los modelos, se ha realizado una prueba con una imagen y se ha medido el tiempo que tarda cada modelo en realizar la predicción.

Es interesante analizar el tiempo de inferencia al desplegar un modelo en un dispositivo de bajo rendimiento, como un dispositivo móvil o un dispositivo IoT.

Para poder visualizar como los datos de validación se distribuyen en un espacio de 2 dimensiones, realicé una reducción de dimensionalidad utilizando el algoritmo t-SNE en las características extraídas por el modelo Xception.

Al observar el gráfico, notamos que las categorías Ropa y Marrón están claramente definidas y separadas. Sin embargo, las categorías Medicamento y Restos parecen menos distintas. Esta repre-

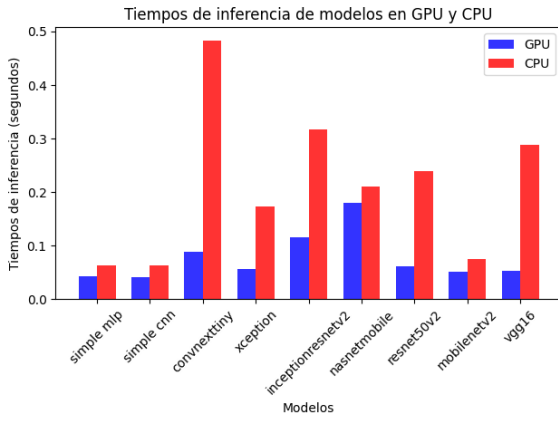


Fig. 4: Tiempos de inferencia de los modelos.



Fig. 5: Visualización de los datos de validación en un espacio de 2 dimensiones.

sentación visual nos sugiere que el clasificador podría confundirse al asignar una observación a una de estas dos últimas clases.

Para confirmar esta sospecha, es recomendable examinar la matriz de confusión del clasificador, que muestra la cantidad de observaciones clasificadas correctamente y las que no lo fueron para cada par de clases. También es útil considerar la tasa de error entre las diferentes clases para identificar posibles problemas en la clasificación.

La matriz de confusión nos permite confirmar que las clases que están cercanas en el espacio de dos dimensiones son las que tienen mayor probabilidad de confundirse, mientras que las clases más separadas tienen menos posibilidades de ser confundidas. Esto se debe a que las clases que están más cercanas en el gráfico pueden tener características similares, lo que hace que el clasificador tenga dificultades para distinguirlas.

5.4. Imágenes similares

Para mejorar la eficacia del sistema de clasificación de imágenes y ayudar al usuario cuando el modelo de clasificación se encuentre en una situación de incertidumbre, se ha implementado un algoritmo de búsqueda de imágenes similares. Este algoritmo, dada una imagen proporcionada por el usuario, devuelve las tres imágenes más parecidas a la imagen de entrada.

El proceso se puede dividir en dos fases esenciales: la extracción de características de la imagen y la búsqueda de imágenes similares.

Amarillo	0.88	0.02	0.00	0.01	0.02	0.01	0.01	0.01	0.01	0.02
Azul	0.04	0.85	0.02	0.00	0.01	0.01	0.02	0.03	0.02	0.00
Marron	0.01	0.01	0.96	0.00	0.00	0.01	0.00	0.01	0.00	0.00
Medicamento	0.03	0.00	0.00	0.90	0.00	0.01	0.00	0.01	0.00	0.06
Pilas	0.01	0.00	0.00	0.00	0.98	0.01	0.00	0.00	0.00	0.00
Punto	0.03	0.01	0.01	0.00	0.01	0.86	0.06	0.01	0.00	0.01
RAEE	0.02	0.00	0.00	0.00	0.01	0.02	0.91	0.02	0.00	0.02
Resta	0.04	0.01	0.00	0.01	0.00	0.03	0.02	0.87	0.01	0.01
Ropa	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.96	0.00
Verde	0.04	0.00	0.00	0.03	0.00	0.00	0.01	0.01	0.00	0.89
	Amarillo	Azul	Marron	Medicamento	Pilas	Punto	RAEE	Resta	Ropa	Verde
	Clase predicha									
	Clase verdadera									

Fig. 6: Visualización la matriz de confusión del modelo Xception con los datos de validación.

Para la extracción de características, se ha utilizado un modelo de red neuronal convolucional llamado Xception. Las redes neuronales convolucionales son especialmente eficaces en la extracción de características relevantes de las imágenes, y Xception es uno de los modelos más eficaces en este ámbito. En nuestro caso, la imagen de entrada tiene una forma de $224 \times 224 \times 3$ (ancho, alto y canales de color respectivamente), y luego se transforma a un vector de características de 1024 dimensiones a través de la red Xception. Este vector se obtiene de la penúltima capa de la red, es decir, se trata de una representación de nivel superior de la imagen, que captura las características más abstractas y relevantes de la imagen.

Una vez obtenido el vector de características, este se utiliza como entrada para un algoritmo de K Nearest Neighbors (KNN). El KNN se encarga de calcular la distancia euclídea entre la representación vectorial de la imagen de entrada y las representaciones de todas las imágenes en el dataset. La fórmula para la distancia euclídea entre dos vectores x y y en un espacio de n dimensiones es la siguiente:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Después de calcular estas distancias, el algoritmo KNN devuelve las tres imágenes cuyos vectores de características son los más cercanos al vector de características de la imagen de entrada. En otras palabras, devuelve las tres imágenes más similares a la imagen proporcionada por el usuario.

Este enfoque proporciona una capa adicional de flexibilidad al sistema de clasificación, permitiendo al usuario seleccionar la etiqueta más adecuada, como se puede observar en la figura 7. En este caso, el usuario ha proporcionado una imagen de un cable (RAEE), y ha devuelto tres imágenes similares de cables. El usuario puede entonces seleccionar la etiqueta más adecuada para la imagen de entrada en caso de no estar de acuerdo con la predicción del modelo de clasificación.

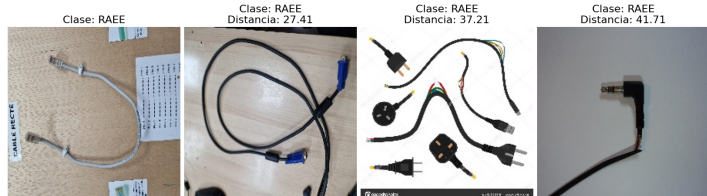


Fig. 7: Visualización de imágenes similares de los datos de validación con imágenes de entrenamiento. La primera imagen es la imagen de entrada, y las tres siguientes son las imágenes más similares.

5.5. CLIP

CLIP es una tecnología con habilidades zero-shot muy potentes y que nos puede ser útil para proporcionar información adicional al usuario sobre la imagen que ha subido y también para clasificar imágenes. Para ello se ha realizado un listado de las clases que quieres clasificar que se encuentran en la pagina ResiduOn-Vas siendo un total de 247 clases.

La implementacion de CLIP que se ha utilizado ha sido OpenCLIP con la variante ViT-g/14.

Estos modelos son altamente sensibles a la formulacion del prompt de salida, por lo que se han realizado pruebas con distintas formulaciones para ver cual es la que mejor se adapta al dataset.

Prompt	acc	std
{ }	0.707	0.158
a photo of { }	0.753	0.140
a picture of { }	0.730	0.171
a photo of a { }	0.760	0.144
a picture of a { }	0.747	0.151
an image of { }	0.759	0.152
an image of a { }	0.762	0.152
a bright photo of a { }	0.745	0.147
a photo of { }, a type of waste	0.761	0.111
a picture of { }, a type of waste	0.760	0.115
a photo of a { }, a type of waste	0.778	0.105
a picture of a { }, a type of waste	0.782	0.106
an image of { }, a type of waste	0.757	0.113
an image of a { }, a type of waste	0.779	0.109
a bright photo of a { }, a type of waste	0.782	0.114
a photo of { }, a type of domestic waste	0.766	0.101
a picture of { }, a type of domestic waste	0.772	0.101
a photo of a { }, a type of domestic waste	0.779	0.096
a picture of a { }, a type of domestic waste	0.783	0.101
an image of { }, a type of domestic waste	0.765	0.107
an image of a { }, a type of domestic waste	0.773	0.099
a bright photo of a { }, a type of domestic waste	0.769	0.103

Tabla 8: Accuracy de los distintos prompts

Tras realizar pruebas con el dataset de validación hemos obtenido que el mejor prompt es *a picture of a label, a type of waste* como se puede observar en la tabla 12.

Si implementamos OpenCLIP como mecanismo de extracción de características y, posteriormente, entrenamos un modelo lineal basándonos en estas características, podemos alcanzar una pre-

cisión (accuracy) de 0.93. Este valor representa el resultado más alto obtenido hasta la fecha en comparación con los otros modelos evaluados.

Además, se puede observar que los modelos que generan buenos resultados en el escenario de aprendizaje zero-shot, aquel en el que el modelo debe inferir sobre datos que nunca ha visto durante el entrenamiento, muestran una gran resistencia frente a la utilización de conjuntos de entrenamiento más reducidos. En otras palabras, aún cuando se disponga de una cantidad limitada de datos para el entrenamiento, estos modelos mantienen un rendimiento notable.

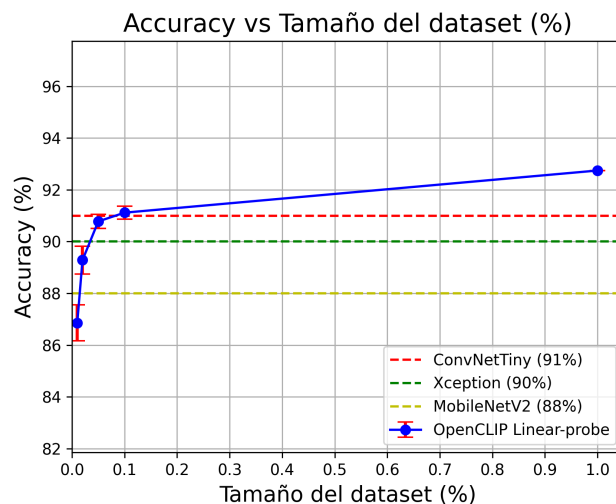


Fig. 8: Accuracy en funcion del numero de imagenes de entrenamiento.

La Figura 8 compara el rendimiento de varios modelos de aprendizaje automático, incluyendo Linear-probe OpenCLIP, Xception y MobilenetV2. La métrica de rendimiento en este caso es del accuracy, que es una medida de cuántas veces el modelo hace predicciones correctas.

Lo que es interesante de esta figura es que muestra que el modelo Linear-probe OpenCLIP es capaz de obtener una precisión de 0.91 % con tan solo el 10 % del conjunto de datos de entrenamiento. Este es un resultado impresionante y muestra la eficiencia de este modelo. En la mayoría de los casos, necesitamos un gran volumen de datos para entrenar un modelo de aprendizaje automático para que sea efectivo, pero el Linear-probe OpenCLIP parece ser capaz de “aprender” mucho de un conjunto de datos más pequeño.

Esto contrasta con los modelos Xception y MobilenetV2, que pueden no haber obtenido resultados tan buenos. Estos son modelos preentrenados conocidos que se usan con frecuencia en tareas de clasificación de imágenes y son muy eficaces en muchos casos. Sin embargo, parece que en esta instancia, no pudieron superar a OpenCLIP cuando se usaron los mismos datos.

En la tabla 10, se ha analizado y comparado la eficacia de varias técnicas de clasificación de residuos, incluyendo la aplicación Aire, junto con varios modelos de aprendizaje profundo como Xception, MobileNetV2, OpenCLIP Zero-Shot y OpenCLIP Linear-probe. Los modelos se han comparado en base a su precisión media en la clasificación de diferentes tipos de residuos: Amarillo, Azul, Marrón, Médica, Pilas, Punto, RAEE, Resta, Ropa y Verde.

	Amarillo	Azul	Marron	Medica	Pilas	Punto	RAEE	Resta	Ropa	Verde	Total acc
Xception	0.88	0.85	0.96	0.9	0.98	0.86	0.91	0.87	0.96	0.89	0.9
mobilenetV2	0.9	0.84	0.97	0.83	0.97	0.81	0.86	0.84	0.94	0.87	0.88
Zero Shot OpenCLIP	0.79	0.67	0.76	0.67	0.94	0.82	0.95	0.64	0.8	0.78	0.78
Linear-probe OpenCLIP	0.92	0.91	0.97	0.9	0.99	0.89	0.95	0.88	0.99	0.9	0.93

Tabla 9: Comparativa de los modelos con los datos de validación.

	Amarillo	Azul	Marron	Medica	Pilas	Punto	RAEE	Resta	Ropa	Verde	Mean Acc
Aire	0.15	0.10	0.40	0.00	0.10	0.30	0.00	0.00	0.00	0.00	0.10
Aire + Material	0.75	0.30	0.40	0.00	0.10	0.30	0.00	0.00	0.00	0.80	0.27
Xception	0.57	0.44	0.89	0.47	0.75	0.52	0.90	0.41	0.90	0.80	0.67
MobileNetV2	0.67	0.39	0.84	0.06	0.60	0.48	0.85	0.55	0.80	0.75	0.60
OpenCLIP Zero-Shot	1.00	0.55	0.79	0.82	0.90	1.00	0.95	0.91	0.85	0.95	0.87
OpenCLIP Linear-probe	0.90	0.55	0.95	0.60	0.95	0.60	0.70	0.60	0.75	0.80	0.73

Tabla 10: Comparativa de los modelos desarrollados con la aplicación Aire.

La versión base de la aplicación Aire mostró un rendimiento considerablemente más bajo en comparación con los otros modelos, con una precisión media de sólo el 10 %. Sin embargo, cuando se le añadió información adicional sobre los materiales (Aire + Material), la precisión media aumentó al 27 %, demostrando una mejora significativa pero aún insuficiente frente a los modelos de aprendizaje profundo.

El modelo Xception, una red convolucional popular en el campo del reconocimiento de imágenes, mostró un rendimiento sólido, alcanzando una precisión media de 67 %. Su contraparte, MobileNetV2, un modelo optimizado para dispositivos móviles con recursos limitados, también demostró un buen desempeño con una precisión media del 60 %.

Por otro lado, los modelos OpenCLIP, una implementación reciente de aprendizaje sin supervisión de OpenAI, superaron a todos los demás modelos en nuestra comparación. En particular, el modelo OpenCLIP Zero-Shot demostró una precisión asombrosa en la clasificación de residuos, alcanzando una precisión media de 87 %. El modelo OpenCLIP Linear-probe también mostró un buen desempeño con una precisión media del 73 %.

Estos resultados sugieren que los enfoques basados en aprendizaje profundo, como los proporcionados por OpenCLIP, pueden ofrecer mejoras sustanciales en la clasificación de residuos en comparación con las soluciones basadas en aplicaciones como Aire. Sin embargo, también sugieren que las mejoras pueden ser posibles a través de la incorporación de información adicional de materiales en la aplicación Aire, aunque este enfoque aún no alcanza la eficacia de los modelos de aprendizaje profundo probados.

Por lo tanto, estos hallazgos apoyan la implementación y el uso de técnicas de aprendizaje profundo en la clasificación de residuos para lograr una mayor precisión y eficiencia en esta tarea crítica.

6 TRABAJO FUTURO

En este trabajo se ha desarrollado una aplicación para clasificar residuos utilizando aprendizaje profundo. Sin embargo, hay muchas formas en las que se puede mejorar la aplicación para que sea más eficaz y útil. Entre ellas se encuentran, la segmentación de objetos

dentro de la misma imagen para poder separar los posibles materiales antes de tirar el residuo. Para ello se ha investigado un poco de como se podría implementar y se han encontrado varias formas de hacerlo.

6.1. Segmentación de objetos

Unos de los principales problemas de utilizar clasificación en imágenes es que no se puede saber la localización de los objetos en la imagen. Para ello es necesario crear un dataset de segmentación, pero el coste en tiempo humano para etiquetar las imágenes es muy elevado. Para solucionar este problema se ha buscado métodos de segmentación utilizando Segment Anything de Meta para poder segmentar y clasificar las imágenes de la manera automática y terminar de etiquetar las imágenes que no se han podido etiquetar correctamente de manera manual. Segment Anything funciona de dos maneras distintas, en la primera se le pasa una imagen y devuelve una imagen con los objetos segmentados, en la segunda se le pasa una imagen y puntos donde se encuentra el objeto y puntos que es el fondo y devuelve una máscara del objeto.

Sabiendo esto se puede afrontar el problema de clasificación automática de diversas formas.

6.1.1. Clasificación de máscaras

Este método consiste en usar Segment Anything en una imagen y que devuelva un listado de máscaras de todos los objetos que se encuentran en la imagen. Una vez se tiene este listado de máscaras se puede pasar a la red neuronal para que clasifique cada una de las imágenes nuevas.

Una vez tenemos las máscaras de los objetos se puede pasar a la red neuronal para que clasifique cada una de las imágenes nuevas y nos quedamos con la clase que mayor porcentaje de confianza tenga.

Este método tiene el problema de confundir objetos o zonas de la imagen que no nos interesa como si fuera el objeto que se está clasificando como se indica en la figura 9. Por lo que es necesario

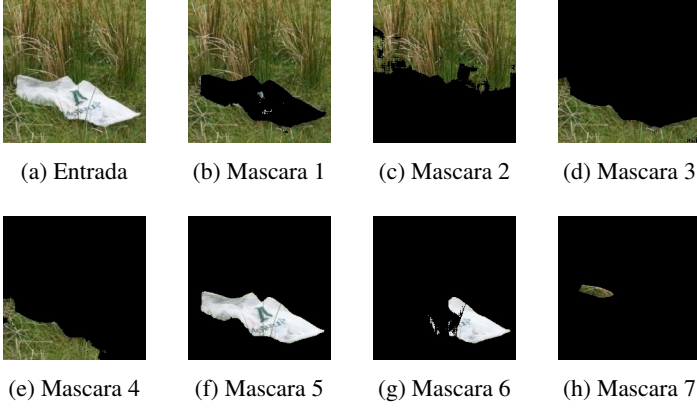


Fig. 9: Imagen de entrada y mascaras de los objetos detectados. La mascara que el clasificador ha escogido ha sido la (g).

descartar aquellas mascaras que no se correspondan con el objeto que se esta clasificando, para ello necesitamos poner un atención visual en la imagen para saber donde se encuentra el objeto que se esta clasificando.

6.1.2. Atención visual para las mascaras

Para seleccionar las macara idónea se puede utilizar la técnica de Saliency map de esta manera podemos saber donde se encuentra el objeto que se esta clasificando y descartar las mascaras que no se encuentren en esa zona.

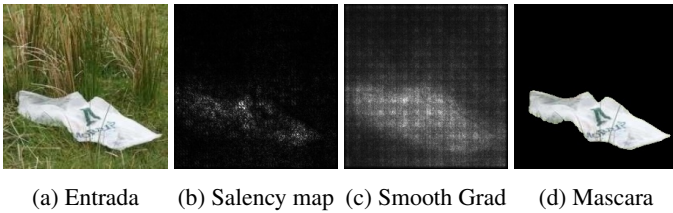


Fig. 10: Comparativa de saliency map y smooth grad para seleccionar la mascara idónea que maximiza los valores del SmoothGrad.

Al inicio cogemos la imagen y realizamos su mapa de saliencia S :

$$S_{\text{mean}} = \frac{1}{N \times M} \sum_{i=1}^N \sum_{j=1}^M S_{i,j}$$

S_{mean} es el promedio de todos los elementos en el mapa de saliencia S . N es el número total de pixeles en S .

Por cada máscara generada M , calculamos el valor final V de la siguiente manera:

1. Calculamos el área de la máscara M .

$$A = \sum_{i=1}^N \sum_{j=1}^M M_{i,j}$$

A es el área de la máscara M y N es el número total de elementos en M .

2. Actualizamos cada elemento $S_{i,j}$ en el mapa de saliencia S .

$$S_{i,j} = \begin{cases} S_{i,j}, & \text{si } S_{i,j} \geq 0,5 \cdot S_{\text{mean}} \\ -2, & \text{de lo contrario} \end{cases}$$

3. Calculamos el valor final V .

$$V = \sum_{i=1}^N (M_{i,j} \cdot S_{i,j})$$

V es el valor final que se calcula como la suma de los productos de los elementos correspondientes en la máscara M y el mapa de saliencia S .

Una vez hemos calculado V en todas las macaras nos quedamos con la que tenga el valor mas alto, ya que esta sera la que se corresponda con el objeto que se esta clasificando.

6.1.3. Optimización de puntos de atención

Como ultimo método tenemos la optimización de puntos de atención, este método consiste en generar una serie de puntos de atención en la imagen y pasarlos a Segment Anything para que genere la mascara de los objetos que se encuentran en esos puntos de atención.

Los parámetros utilizados de Segment Anything son:

```
stability_score_thresh = 0.1
pred_iou_thresh = 0.1
box_nms_thresh = 0.9
```

Y los pasos del algoritmo son los siguientes:

1. LayerCAM se aplica para obtener un mapa de activación de clase A . Se usa la capa de destino final en el modelo y se calculan los gradientes para obtener una matriz 2D.
2. Se calculan las ubicaciones de los valores máximos y mínimos en este mapa de activación:

$$\text{max_coord} = \arg \max(A), \quad \text{min_coord} = \arg \min(A)$$

donde max_coord y min_coord son las coordenadas de los puntos con el valor máximo y mínimo en A respectivamente.

3. Se define una función f que realiza una serie de operaciones:

$$f(\text{coords}) = 1 - \text{Pred}(\text{Mask}(\text{Image}, \text{coords}) \cdot \text{Image})$$

donde coords son las coordenadas de entrada a la función, Mask es la función que aplica la máscara generada por el Segment Anything a la imagen original, e Image es la imagen original. Pred es el modelo pre-entrenado Xception que se utiliza para predecir la clasificación de la imagen. $f(\text{coords})$ devuelve el error entre la predicción del modelo y la clase objetivo.

4. Finalmente, se utiliza el método de “minimize” de scipy para encontrar las coordenadas que minimizan $f(\text{coords})$ dentro de un rango específico para cada coordenada:

$$\text{res} = \min_{0 \leq x, y \leq 224} f([x_{\text{max}}, y_{\text{max}}, x_{\text{min}}, y_{\text{min}}])$$

donde $x_{\text{max}}, y_{\text{max}}$ son las coordenadas del valor máximo, y $x_{\text{min}}, y_{\text{min}}$ son las coordenadas del valor mínimo en A . res es el resultado de la optimización, y son las coordenadas que minimizan $f(\text{coords})$.

Es importante inicial-izar los puntos con los valores máximos y mínimos en A para que el algoritmo pueda encontrar los puntos de atención que maximizan la activación de la clase objetivo, ya que si no se hace acabaremos en un mínimo local.

6.1.4. Comparativa visual de los métodos

Las pruebas se han realizado con 8 imágenes del dataset de entrenamiento escogidas aleatoria-mente.




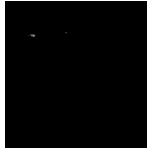






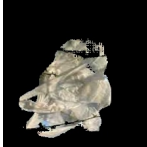
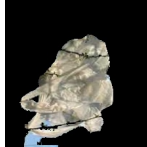


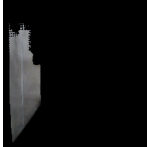




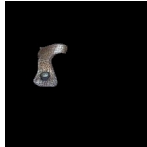



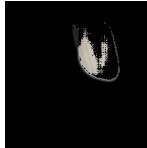


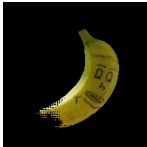
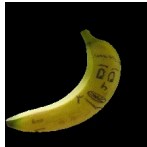




Imagen	Clasificación	Salency	Optimización
			
			
			
			
			
			
			
			

Tabla 11: Comparativa visual de las distintas técnicas para generar mascarar del objeto. En la primera columna tenemos la imagen original, en la segunda el método de clasificación de máscara, en la tercera método de Salency Map y en la cuarta el método de optimización de puntos.

Como resultado se ha obtenido que el método de Salency Map es el mas robusto.

7 CONCLUSIONES

En base a los objetivos planteados y los resultados obtenidos en este estudio, puedo concluir lo siguiente:

Primero, se ha logrado generar un robusto conjunto de datos, compuesto por más de 48000 imágenes, para cubrir una amplia gama de residuos reciclables y no reciclables. Este logro fue posible, en parte, mediante el uso de una aplicación móvil para permitir un etiquetado colaborativo, ampliando considerablemente la diversidad y cantidad de los datos recolectados.

Segundo, se ha desarrollado una aplicación móvil capaz de superar a las aplicaciones existentes en la clasificación automatizada de residuos. En comparación con Aire, la única otra aplicación a nivel nacional domestica que acepta imágenes, mi aplicación demostró una superioridad significativa. Esta mejora facilitará el proceso de reciclaje para los usuarios, permitiéndoles clasificar de manera más eficiente sus residuos.

Tercero, he explorado la viabilidad de generar un conjunto de datos semánticos a partir de uno de clasificación. A pesar de que WSSS mostró sus límites y la complejidad inherente de pasar de un conjunto de datos de imágenes a uno de segmentación de manera no supervisada, esta exploración proporcionó ha proporcionado información valiosa sobre el potencial de las técnicas de segmentación no supervisada para la clasificación de residuos.

Por último, en cuanto a las técnicas de aprendizaje automático, encontramos que OpenCLIP proporcionó los mejores resultados en las tareas. En particular, ha demostrado ser altamente efectivo en tareas de zero-shot, dado la variedad de imágenes que pueden presentarse en los residuos.

AGRADECIMIENTOS

Quiero agradecer sobretodo a mi tutor, Coen Antens, por dejarme total libertad durante el desarrollo de este proyecto. También quiero agradecer a todos aquellos que han participado en el etiquetado de las imágenes, en especial mis alumnos que son los que mas han participado durante el etiquetado con la aplicación móvil.

REFERENCIAS

- [1] Por que es importante reciclar <https://ecoembesdudasreciclaje.es/por-que-es-importante-reciclar/>
- [2] Green Deal https://commission.europa.eu/strategy-and-policy/priorities20192024/europeangreendeal_es
- [3] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2021). High-Resolution Image Synthesis with Latent Diffusion Models. doi:10.48550/ARXIV.2112.10752
- [4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention Is All You Need. doi:10.48550/ARXIV.1706.03762
- [5] GATO (A genelist Agent) <https://openreview.net/pdf?id=1ikK0kHjvj>
- [6] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... Amodei, D. (2020). Language Models are Few-Shot Learners. doi:10.48550/ARXIV.2005.14165
- [7] Language Models are Unsupervised Multitask Learners <https://d4mucfpksyww.cloudfront.net/better-language-models/language-models.pdf>
- [8] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision. doi:10.48550/ARXIV.2103.00020
- [9] PaddleOCR <https://github.com/PaddlePaddle/PaddleOCR>
- [10] Tensorflow <https://www.tensorflow.org/>
- [11] Google Colab <https://colab.research.google.com/>
- [12] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1800-1807). <https://doi.org/10.1109/CVPR.2017.195>
- [13] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity Mappings in Deep Residual Networks. arXiv preprint arXiv:1603.05027.
- [14] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2018). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).
- [15] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ... Adam, H. (2019). Searching for MobileNetV3. arXiv preprint arXiv:1905.02244.
- [16] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. (2019). MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv preprint arXiv:1801.04381.
- [17] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision. arXiv preprint arXiv:1512.00567.
- [18] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Thirty-First AAAI Conference on Artificial Intelligence.
- [19] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A ConvNet for the 2020s. arXiv preprint arXiv:2201.03545.
- [20] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning Transferable Architectures for Scalable Image Recognition. arXiv preprint arXiv:1707.07012.
- [21] Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.
- [22] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... Girshick, R. (2023). Segment Anything. arXiv preprint arXiv:2304.02643.

APÉNDICE

A.1. Aplicación móvil

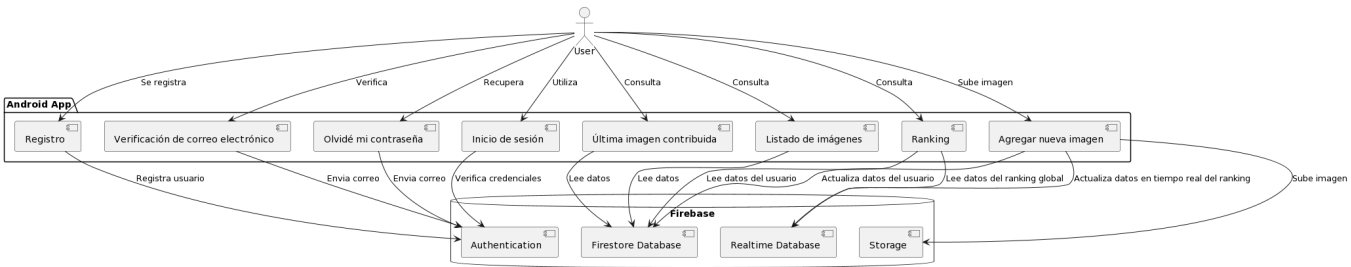


Fig. 11: Diagrama de componentes de la aplicación Android para contribuir y visualizar imágenes en tiempo real utilizando Firebase como backend.

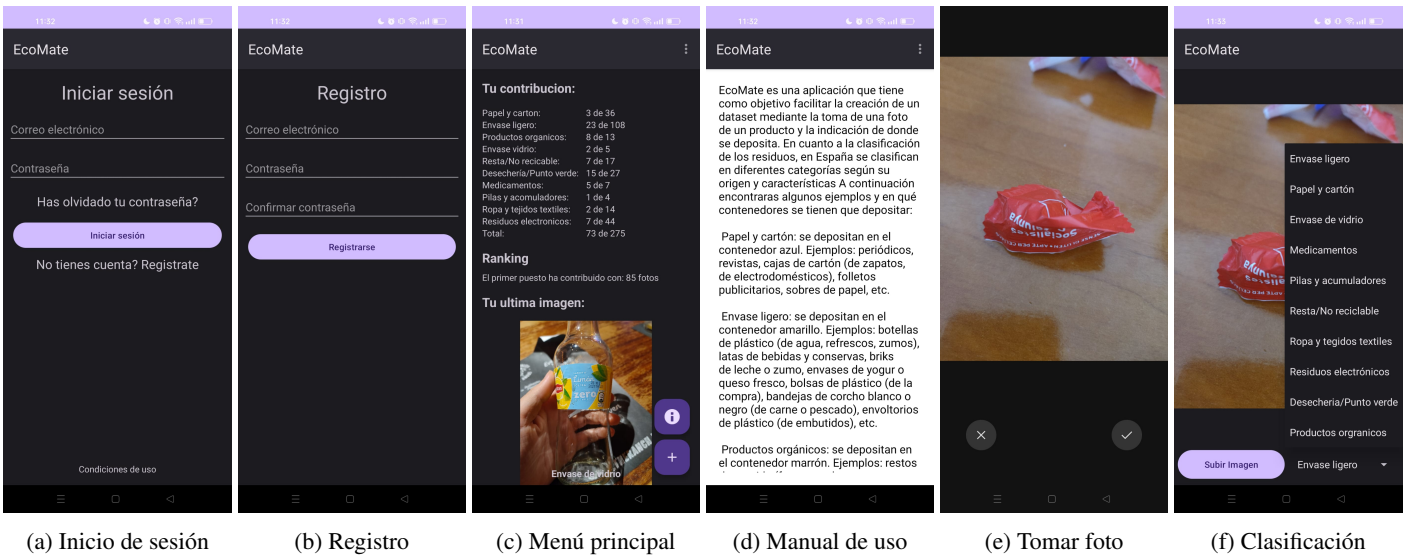


Fig. 12: Interfaz de la aplicación Android.

A.2. Modelos de clasificación

Para la red neuronal simple_mlp se utilizó la arquitectura mostrada en la tabla 13.

Para la red neuronal simple_cnn se utilizó la arquitectura mostrada en la tabla 14.

Para las redes convolucionales pre-entrenadas se ha usado la misma arquitectura, solo ha cambiado el modelo convolucional base. La arquitectura de las redes convolucionales se encuentra en la tabla 15.

A.3. Prompts para OpenCLIP

Los prompts usados para OpenCLIP se encuentran en la tabla 16.

Model	Train acc	Train loss	Val acc	Val loss	Params	Epochs	Time (s)/Epoch	Time (s)
convnexttiny_no_freeze	0.57	1.17	0.91	0.36	66.37M	12	1799.02	21588.27
xception_fine_no_freeze	0.62	1.07	0.90	0.36	123.63M	10	670.20	6702.02
xception_fine_no_freeze_no_aug	0.99	0.02	0.90	0.53	123.63M	6	808.19	4849.13
inceptionresnetv2_no_freeze	0.65	0.98	0.90	0.39	93.67M	12	1306.87	15682.44
inceptionv3_no_freeze	0.56	1.20	0.89	0.42	74.24M	19	684.87	13012.54
densenet121_no_freeze	0.55	1.22	0.89	0.42	58.43M	13	1050.03	13650.33
mobilenetv2_no_freeze_no_aug	1.00	0.02	0.88	0.70	66.49M	10	335.72	3357.20
mobilenetv2_no_freeze	0.58	1.15	0.88	0.44	66.49M	15	358.09	5371.32
nasnetmobile_no_freeze	0.60	1.10	0.87	0.54	57.27M	13	789.63	10265.25
resnet50v2_f_no_freeze	0.55	1.22	0.86	0.54	126.34M	17	649.03	11033.57
mobilenetv3large_no_freeze	0.56	1.20	0.86	0.52	51.18M	18	352.28	6341.13
vgg16_no_freeze	0.54	1.25	0.84	0.56	40.42M	22	689.81	15175.75
resnet50_no_freeze	0.55	1.22	0.84	0.57	126.36M	14	698.19	9774.64
xception	0.93	0.22	0.83	0.69	123.63M	5	280.99	1404.95
inceptionresnetv2	0.92	0.23	0.83	0.78	93.67M	6	510.01	3060.03
nasnetmobile	0.92	0.26	0.81	0.73	57.27M	4	263.30	1053.21
densenet121	0.91	0.28	0.81	0.76	58.43M	4	296.39	1185.57
resnet50v2	0.93	0.24	0.81	1.10	126.34M	4	291.10	1164.40
mobilenetv2	0.95	0.18	0.80	1.06	66.49M	4	114.87	459.49
mobilenetv3small_no_freeze	0.54	1.25	0.80	0.74	29.85M	24	201.78	4842.67
inceptionv3	0.93	0.21	0.80	0.92	74.24M	6	276.80	1660.82
nasnetlarge_aug	0.92	0.24	0.79	1.02	287.24M	5	1025.08	5125.40
xception_aug	0.65	1.02	0.75	0.84	123.63M	14	280.99	3933.85
vgg16_pre	0.96	0.12	0.74	1.36	40.42M	8	221.76	1774.11
vgg19	0.91	0.27	0.71	1.34	45.73M	6	228.42	1370.51
vgg16_pre_no_freeze	0.65	0.96	0.59	1.48	40.42M	9	727.05	6543.48
simple_cnn	0.97	0.11	0.54	3.78	44.4M	7	184.61	1292.26
convnexttiny	0.73	0.77	0.53	1.67	66.37M	6	727.44	4364.65
convnextsmall	0.66	1.01	0.52	1.61	88.0M	10	1164.44	11644.40
mobilenetv3large	0.51	1.43	0.42	1.76	51.18M	13	126.94	1650.24
resnet50	0.51	1.44	0.35	1.93	126.36M	15	309.29	4639.32
mobilenetv3small	0.37	1.80	0.34	1.89	29.85M	14	84.99	1189.87
simple_mlp	0.38	1.76	0.33	1.95	77.24M	22	71.35	1569.77
efficientnetv2b0	0.15	2.27	0.11	2.31	70.16M	4	194.60	778.41
vgg19_no_freeze	0.15	2.27	0.11	2.31	45.73M	8	774.15	6193.17

Tabla 12: Resultados de la evaluación de los modelos de redes neuronales.

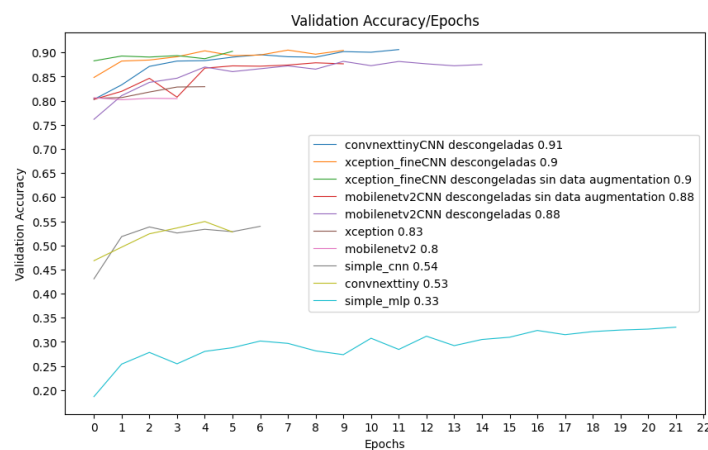


Fig. 13: Gráfica del accuracy por épocas de los modelos de redes neuronales.

Capa	Tipo	Parámetros
Entrada	Aplanar (Flatten)	(224, 224, 3)
Capa Densa 1	Densa (Dense)	512, activación='relu'
Capa Densa 2	Densa (Dense)	256, activación='relu'
Capa Densa 3	Densa (Dense)	128, activación='relu'
Capa de Salida	Densa (Dense)	10, activación='softmax'

Tabla 13: Arquitectura de la red neuronal simple_mlp

Capa	Tipo	Parámetros
Entrada	Convolución 2D	32, (3, 3), activación='relu', input_shape=(224, 224, 3)
Capa Max-Pooling 1	MaxPooling2D	2, 2
Capa Convolución 2	Convolución 2D	64, (3, 3), activación='relu'
Capa Max-Pooling 2	MaxPooling2D	2, 2
Capa Convolución 3	Convolución 2D	128, (3, 3), activación='relu'
Capa Max-Pooling 3	MaxPooling2D	2, 2
Capa Aplanar	Aplanar (Flatten)	-
Capa Densa 1	Densa (Dense)	512, activación='relu'
Capa de Salida	Densa (Dense)	10, activación='softmax'

Tabla 14: Arquitectura de la red neuronal simple_cnn

Capa	Tipo	Parámetros
Base ConvNetwork	ConvNetwork	ConvNetwork
Capa Aplanar	Aplanar (Flatten)	-
Capa Densa 1	Densa (Dense)	1024, activación='relu'
Capa de Salida	Densa (Dense)	10, activación='softmax'

Tabla 15: Arquitectura usada para las distintas redes convolucionales.

Prompt	Amarillo	Azul	Marron	Medica	Pilas	Punto	RAEE	Resta	Ropa	Verde
{}	0.78	0.63	0.53	0.44	0.98	0.76	0.93	0.70	0.61	0.70
a photo of {}	0.80	0.64	0.62	0.48	0.98	0.79	0.94	0.75	0.80	0.73
a picture of {}	0.77	0.60	0.42	0.49	0.98	0.80	0.95	0.70	0.79	0.79
a photo of a {}	0.83	0.62	0.70	0.50	0.97	0.81	0.95	0.65	0.81	0.77
a picture of a {}	0.83	0.60	0.59	0.48	0.98	0.81	0.95	0.67	0.76	0.80
an image of {}	0.81	0.56	0.79	0.44	0.98	0.79	0.93	0.79	0.80	0.71
an image of a {}	0.86	0.58	0.76	0.46	0.97	0.82	0.96	0.64	0.78	0.80
a bright photo of a {}	0.84	0.66	0.70	0.47	0.96	0.80	0.95	0.59	0.73	0.75
a photo of {}, a type of waste	0.77	0.67	0.63	0.66	0.94	0.78	0.96	0.63	0.80	0.77
a picture of {}, a type of waste	0.77	0.67	0.60	0.69	0.94	0.80	0.95	0.62	0.79	0.77
a photo of a {}, a type of waste	0.80	0.67	0.74	0.65	0.95	0.81	0.95	0.64	0.80	0.77
a picture of a {}, a type of waste	0.81	0.68	0.72	0.67	0.95	0.82	0.95	0.63	0.80	0.80
an image of {}, a type of waste	0.77	0.64	0.74	0.58	0.90	0.80	0.96	0.63	0.80	0.77
an image of a {}, a type of waste	0.81	0.67	0.78	0.63	0.93	0.81	0.96	0.62	0.81	0.78
a bright photo of a {}, a type of waste	0.83	0.69	0.82	0.66	0.93	0.84	0.97	0.58	0.80	0.73
a photo of {}, a type of domestic waste	0.77	0.67	0.66	0.67	0.91	0.79	0.97	0.69	0.81	0.74
a picture of {}, a type of domestic waste	0.77	0.67	0.66	0.71	0.94	0.80	0.96	0.65	0.80	0.76
a photo of a {}, a type of domestic waste	0.80	0.67	0.73	0.68	0.92	0.81	0.96	0.67	0.81	0.76
a picture of a {}, a type of domestic waste	0.79	0.67	0.76	0.67	0.94	0.82	0.95	0.64	0.80	0.78
an image of {}, a type of domestic waste	0.78	0.64	0.73	0.60	0.89	0.80	0.97	0.67	0.81	0.77
an Image of a {}, a type of domestic waste	0.81	0.65	0.76	0.65	0.91	0.80	0.95	0.65	0.81	0.75
a bright photo of a {}, a type of domestic..	0.81	0.69	0.73	0.67	0.89	0.82	0.97	0.60	0.81	0.71

Tabla 16: Resultados de los distintos prompts para el modelo OpenCLIP, con los datos de validación.