.UBAfiuba
FACULTAD DE INGENIERÍA

Automatización del Etiquetado de Productos mediante Transformadores de Visión

Visión por Computadora III

Carrera de Especialización en Inteligencia Artificial

Autores:

Abril Noguera
Pedro Lucas Barrera
Lautaro Gabriel Medina

Ciudad de Buenos Aires, diciembre de 2025

Resumen

El proyecto desarrollado consiste en la construcción de un sistema capaz de identificar y asignar atributos descriptivos a productos a partir de sus imágenes mediante un modelo de aprendizaje profundo basado en transformadores de visión. El sistema aborda una necesidad concreta del comercio digital, donde el etiquetado manual de artículos representa una tarea costosa, lenta y con alta probabilidad de errores. La solución propuesta automatiza este proceso, mejora la calidad del catálogo y permite acelerar la incorporación de nuevos productos en plataformas de venta en línea.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción general

El presente capítulo introduce el contexto general del proyecto, expone el problema que motivó su desarrollo, presenta el caso de negocio y describe los datos utilizados. Asimismo, se detallan los objetivos del trabajo y su alcance, estableciendo el marco conceptual necesario para comprender las decisiones técnicas y metodológicas que se desarrollan en los capítulos posteriores.

1.1. Contexto y motivación

El crecimiento del comercio electrónico y la disponibilidad masiva de catálogos digitales han incrementado la necesidad de organizar y clasificar grandes volúmenes de imágenes de productos. En este escenario, la automatización del etiquetado visual se ha convertido en una herramienta fundamental para mejorar la eficiencia operativa y garantizar la consistencia en la gestión de catálogos.

El proyecto desarrollado surgió a partir de esta necesidad: se implementó un sistema de etiquetado automático de productos basado en modelos de visión por computadora. Dicho sistema permitió reducir la dependencia del etiquetado manual, disminuir errores humanos y acelerar el procesamiento de catálogos. Esta iniciativa se orientó a evaluar la utilidad de arquitecturas modernas como los *Vision Transformers* en problemas de clasificación multi-etiqueta.

1.2. Caso de negocio

El crecimiento acelerado del comercio electrónico ha generado catálogos con miles de productos que requieren actualización constante. En este contexto, la correcta clasificación y etiquetado de imágenes es un proceso crítico: determina cómo los productos se muestran, cómo se encuentran mediante búsquedas internas y cómo son recomendados por los motores de recomendación. Sin embargo, este proceso suele realizarse de manera manual, lo que introduce varias limitaciones operativas.

En primer lugar, el etiquetado manual implica costos elevados en horas-hombre, particularmente en empresas que gestionan catálogos dinámicos donde ingresan cientos o miles de productos nuevos por semana. En segundo lugar, este proceso presenta altos niveles de inconsistencia debido a la subjetividad de los operadores: productos similares pueden recibir etiquetas distintas o incompletas, lo que perjudica la calidad del catálogo. Además, el tiempo requerido para clasificar grandes volúmenes genera cuellos de botella que ralentizan el lanzamiento de nuevos productos.

Las consecuencias de un etiquetado deficiente se reflejan en múltiples áreas del negocio. Un producto mal clasificado puede no aparecer en las búsquedas relevantes, reducir su tasa de conversión o ser excluido de sistemas automáticos de recomendación, impactando directamente en ventas. Asimismo, un catálogo inconsistente genera fricción en la navegación del usuario, lo que disminuye la satisfacción y deteriora la percepción de calidad del sitio.

En este contexto, resulta necesario contar con un sistema automático capaz de identificar atributos visuales de manera rápida, coherente y escalable. El presente proyecto se enmarca en esta problemática, evaluando la capacidad de modelos basados en *Vision Transformers* para realizar un etiquetado automático y confiable de productos de moda a partir de sus imágenes.

1.3. Propuesta de valor

El sistema desarrollado busca reemplazar o complementar el etiquetado manual mediante un modelo de visión por computadora capaz de asignar automáticamente atributos clave como categoría, tipo, color y género del producto. Esta automatización agrega valor en distintos niveles:

- **Reducción de costos operativos:** Disminuye la necesidad de intervención humana, especialmente en etapas iniciales de carga masiva de catálogo.
- **Consistencia y estandarización:** El modelo aplica criterios homogéneos sin variación entre operadores, reduciendo errores y ambigüedades.
- **Mayor velocidad de procesamiento:** La clasificación automática permite acelerar el tiempo desde la recepción del producto hasta su publicación en el catálogo.
- **Mejor experiencia de usuario:** Catálogos coherentes mejoran las búsquedas, la navegación y la relevancia de las recomendaciones.
- **Escalabilidad:** El sistema puede procesar miles de imágenes sin aumentar el costo marginal, algo imposible con procesos manuales.

En conjunto, la propuesta de valor consiste en un pipeline automatizado capaz de fortalecer la calidad del catálogo digital y optimizar procesos internos, alineándose con prácticas modernas de comercio electrónico basadas en datos y automatización inteligente.

1.4. Objetivos y alcance

El proyecto tuvo como objetivo general desarrollar un sistema capaz de etiquetar automáticamente imágenes de productos de moda utilizando modelos basados en *Vision Transformers*. Este sistema buscó reproducir y estandarizar el proceso de etiquetado que habitualmente se realiza de manera manual, evaluando su capacidad para predecir atributos clave del catálogo tales como categoría, tipo, color y género del producto.

En términos más específicos, el trabajo se propuso:

- Construir un pipeline reproducible de procesamiento de datos que incluyera la descarga, validación y limpieza del dataset original.

- Implementar un modelo de clasificación multi-etiqueta basado en *Vision Transformers*, adaptado a las características del dataset.
- Entrenar y validar el modelo mediante particiones estratificadas, asegurando una evaluación equilibrada de las distintas clases.
- Analizar el desempeño del modelo utilizando métricas estandarizadas como *accuracy*, F1 y *mean Average Precision* (mAP).
- Examinar los errores y sesgos presentes en las predicciones, identificando limitaciones del enfoque.
- Desarrollar una herramienta de inferencia que permitiera aplicar el modelo a nuevas imágenes en un entorno práctico.

El alcance del proyecto se limitó al análisis y desarrollo de un prototipo funcional entrenado sobre un dataset público. No se abordaron aspectos propios de un sistema productivo, tales como el entrenamiento continuo, la integración con plataformas de comercio electrónico, la optimización de tiempos de inferencia para altos volúmenes ni la incorporación de retroalimentación humana en el ciclo de etiquetado. Tampoco se realizaron experimentos con arquitecturas alternativas ni con técnicas de aumento extensivo de datos debido a restricciones de tiempo y recursos computacionales.

A pesar de estas limitaciones, el desarrollo realizado permite demostrar la viabilidad de automatizar el etiquetado de productos mediante modelos de visión por computadora modernos, sentando las bases para futuras mejoras orientadas a escenarios reales de operación.

1.5. Datos utilizados

Para el desarrollo del sistema se utilizó el dataset público *Fashion Product Images (Small)*, disponible en Kaggle[1]. Este conjunto incluye más de 44.000 imágenes de productos de moda junto con un archivo tabular `styles.csv` que contiene información estructurada del catálogo.

El dataset original presenta atributos como categoría general del producto, subcategoría, tipo, género, temporada y color dominante. Cada imagen está identificada mediante un `productId`, lo que permite vincularla con sus metadatos tabulares. Antes de poder utilizarse en el pipeline de entrenamiento, se aplicaron procesos de limpieza, normalización y verificación de integridad debido a la presencia de valores faltantes, clases poco representadas y rutas inconsistentes.

A partir de este conjunto inicial, se construyó un dataset final que incluyó: (i) imágenes validadas y preprocesadas, (ii) atributos seleccionados para la predicción y (iii) una partición estratificada en *train*, *validation* y *test*. Esta estructura permitió evaluar de manera rigurosa el desempeño del modelo entrenado bajo condiciones reales.

[1] https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-small

Capítulo 2

Análisis Exploratorio de Datos

El presente capítulo describe el análisis exploratorio (EDA) realizado sobre el conjunto de datos utilizado para el entrenamiento del sistema de etiquetado automático de productos. El objetivo del EDA es comprender la estructura del dataset, evaluar la calidad de las imágenes, identificar patrones relevantes y detectar posibles problemas que condicionen las decisiones de preprocesamiento y modelado. A partir de este análisis se establecen los requerimientos y transformaciones necesarias para garantizar un entrenamiento adecuado del modelo basado en *Vision Transformers*.

2.1. Descripción general del dataset

El proyecto emplea el dataset público *Fashion Product Images (Small)*, disponible en Kaggle, que contiene más de 44.000 imágenes de productos de moda junto con metadatos tabulares provistos en el archivo `styles.csv`. Cada imagen está asociada a un identificador único (`id`) y cuenta con atributos descriptivos como:

- **masterCategory**: categoría general del artículo.
- **subCategory**: categoría específica intermedia.
- **articleType**: tipo concreto de producto.
- **gender**: público objetivo.
- **baseColour**, **season**, **usage**, entre otros.

Durante el EDA se realizó una verificación de integridad del archivo tabular y una validación de la existencia y apertura de cada imagen asociada.

2.2. Análisis de valores faltantes

La figura 2.1 muestra la proporción de valores faltantes por atributo. Las variables con mayor incompletitud corresponden a `usage`, `season` y `baseColour`, ninguna de las cuales forma parte de los atributos objetivo del proyecto. Los campos esenciales para la clasificación (`gender`, `masterCategory`, `subCategory` y `articleType`) no presentan valores faltantes.

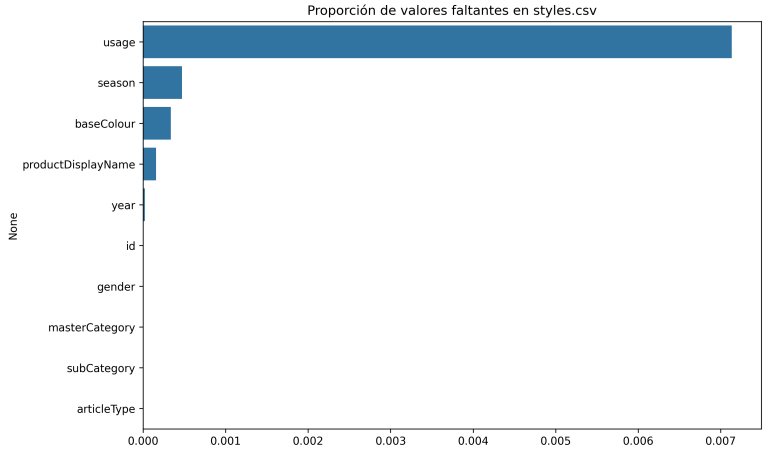

FIGURA 2.1. Proporción de valores faltantes en el archivo `styles.csv`.

Conclusión: No es necesario realizar imputación compleja; el dataset es adecuado para construir las etiquetas objetivo sin modificaciones adicionales.

2.3. Distribución de clases

La figura 2.2 ilustra la distribución de clases para `articleType`. Se observa un marcado desbalance, con categorías como *Tshirts*, *Shirts* y *Casual Shoes* concentrando una cantidad significativamente mayor de muestras respecto de categorías minoritarias.

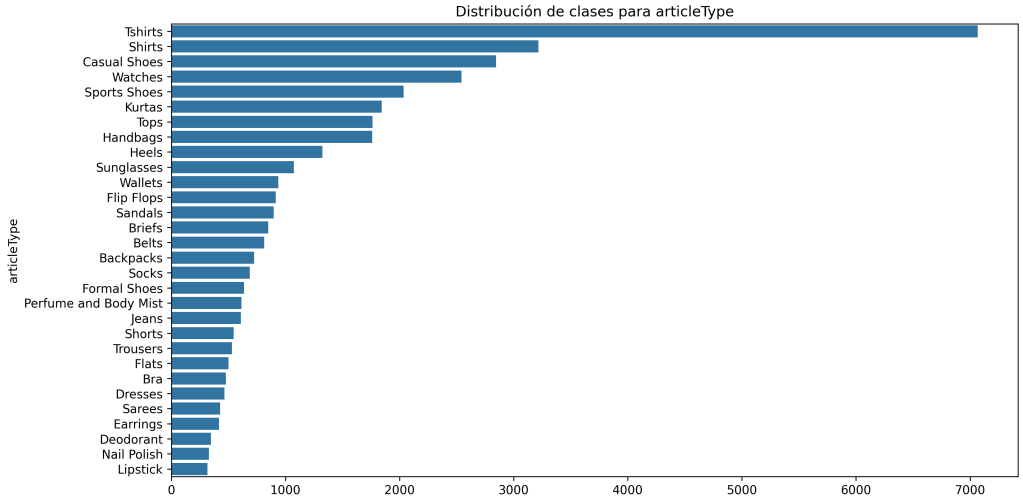

FIGURA 2.2. Distribución de clases para `articleType`.

Patrones similares se observan en `subCategory` y `masterCategory` (figuras 2.3 y 2.4), donde predominan *Topwear*, *Shoes* y *Apparel*, respectivamente.

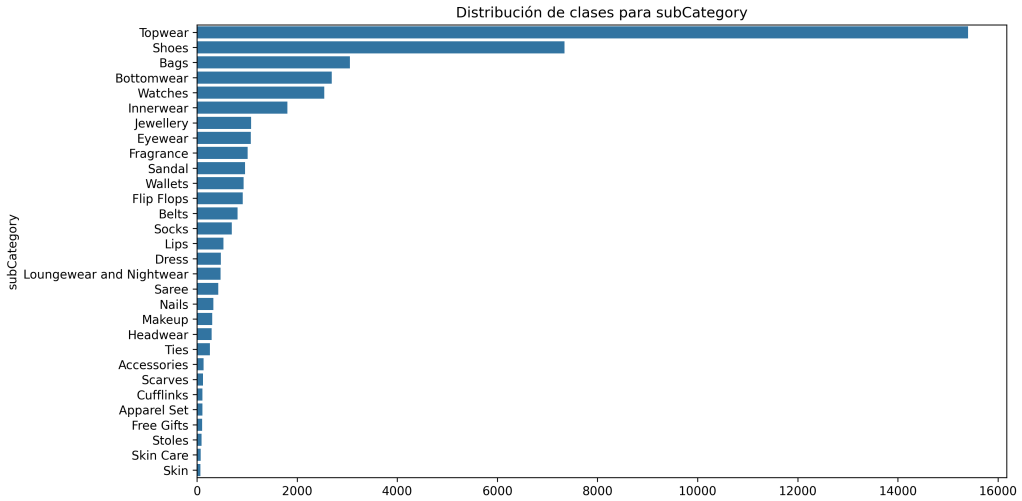

FIGURA 2.3. Distribución de clases para `subCategory`.

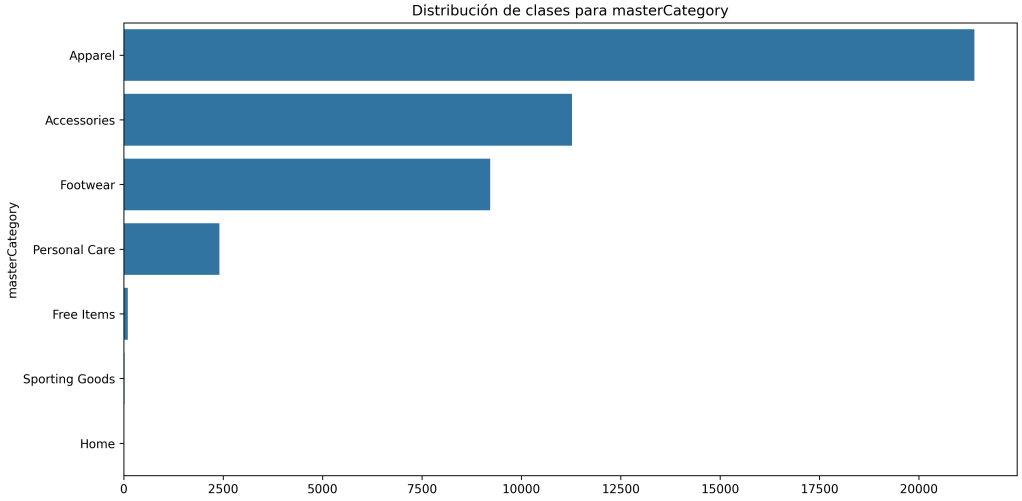

FIGURA 2.4. Distribución de clases para `masterCategory`.

En cuanto al atributo `gender`, la distribución está dominada por *Men* y *Women*, mientras que *Boys*, *Girls* y *Unisex* representan una proporción menor (figura 2.5).

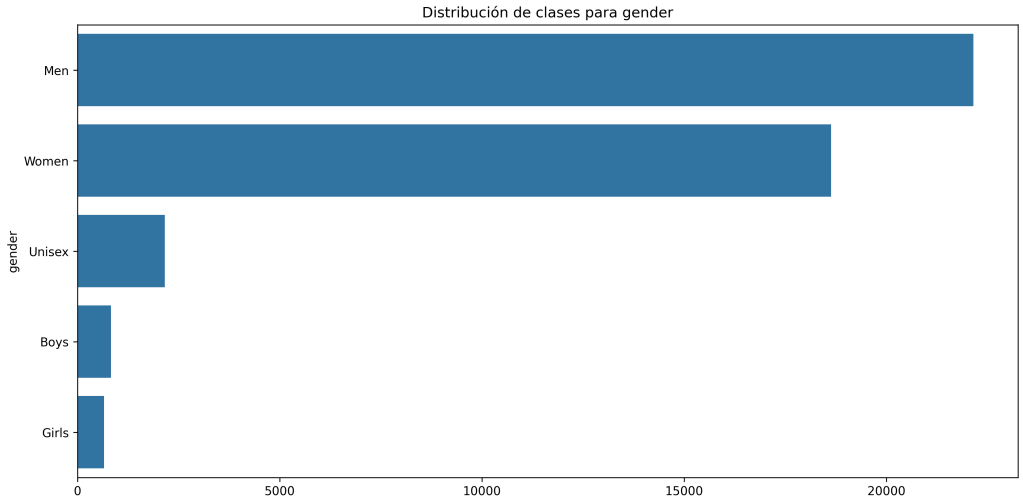

FIGURA 2.5. Distribución de clases para `gender`.

Conclusión: El dataset presenta un fuerte desbalance de clases que puede afectar el aprendizaje del modelo. Esto justifica la aplicación posterior de estrategias como *class weighting*, muestreo estratificado o técnicas de reponderación durante el entrenamiento.

2.4. Inspección visual de muestras

La figura 2.6 muestra una selección aleatoria de imágenes del dataset. Se observa que la mayoría de los productos están fotografiados sobre fondo blanco, correctamente centrados y con iluminación homogénea. La ausencia de fondos complejos y ruido visual simplifica el aprendizaje del modelo.

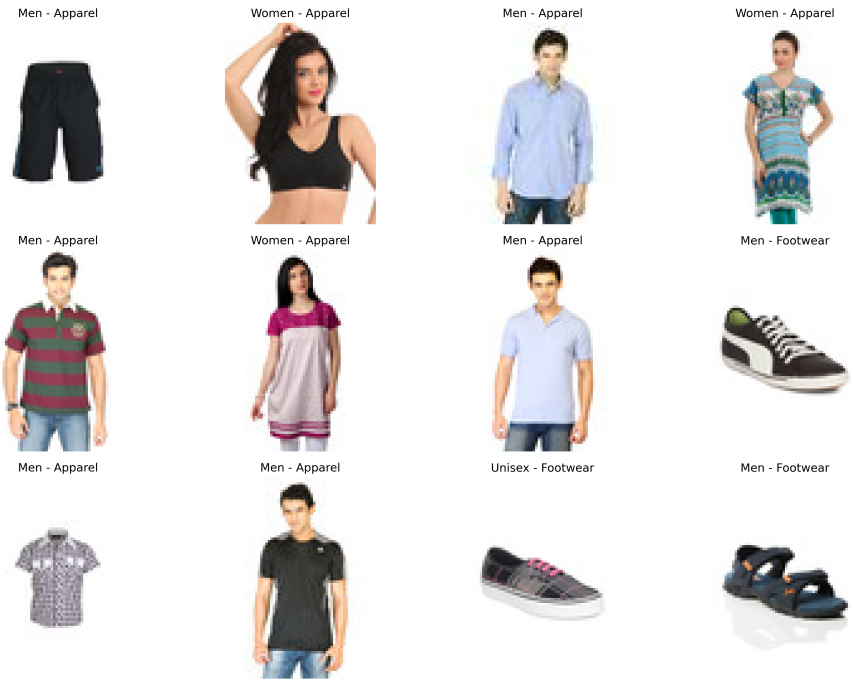

FIGURA 2.6. Muestras aleatorias de imágenes del dataset.

Conclusión: Las condiciones controladas de captura favorecen un desempeño estable en modelos de visión profunda.

2.5. Variabilidad en tamaños de imagen

El análisis de resolución reveló que todas las imágenes tienen un tamaño uniforme de **60×80 px**, con desviación estándar nula. Si bien esta consistencia elimina la necesidad de correcciones geométricas, la resolución es insuficiente para arquitecturas modernas como los *Vision Transformers*, que operan con tamaños estándar de 224×224 px.

Conclusión: Es necesario reescalar todas las imágenes a 224×224 px para compatibilidad con el modelo seleccionado.

2.6. Análisis de calidad de imagen

Se evaluaron métricas complementarias de calidad:

- **Brillo**: valores centrados entre 200 y 230, indicando imágenes bien iluminadas.
- **Contraste**: niveles moderados, consistentes entre productos.
- **Nitidez**: varianza Laplaciana mayormente entre 500 y 4000, suficiente para distinguir bordes.
- **Distribución RGB**: canales balanceados sin dominancia cromática.

Estas mediciones se ilustran en la figura 2.7.

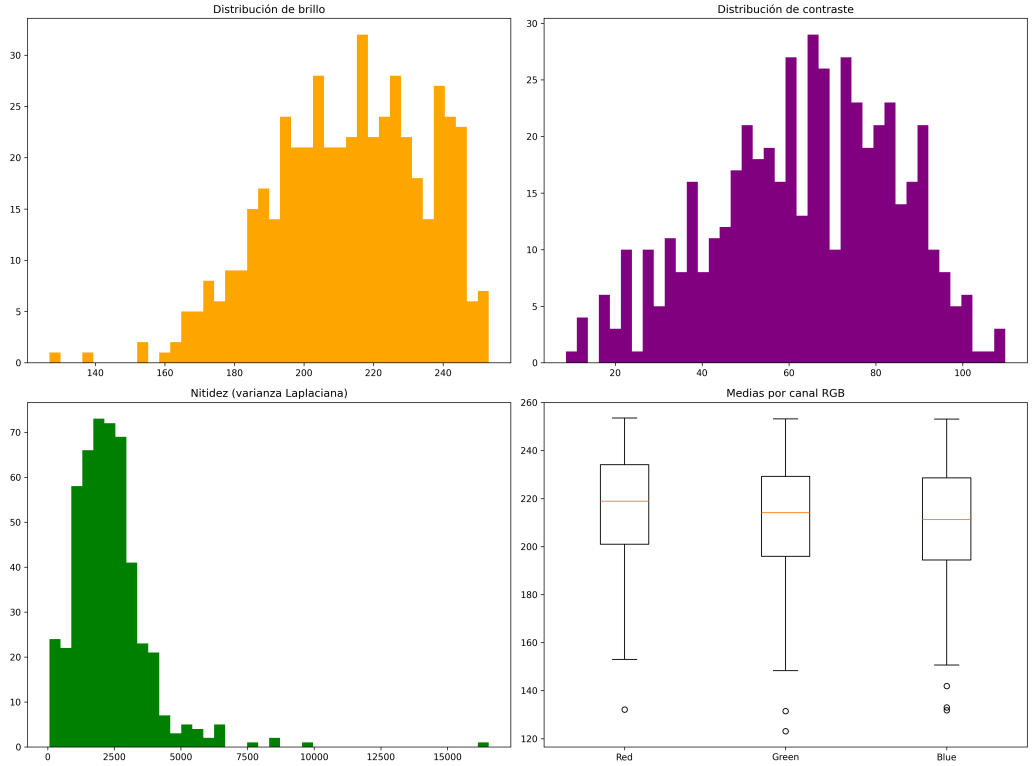

FIGURA 2.7. Análisis de brillo, contraste, nitidez y distribución de color.

Conclusión: La calidad general del dataset es elevada y consistente, lo cual minimiza la necesidad de correcciones avanzadas o normalizaciones fuera del estándar habitual de redes profundas.

2.7. Conclusiones del EDA y su impacto en el modelado

A partir del análisis realizado, se identifican las siguientes decisiones críticas para el pipeline de modelado:

- **Redimensionamiento obligatorio**: todas las imágenes deben ser escaladas a 224×224 px para ser compatibles con *Vision Transformers*.
- **Normalización estándar**: dada la estabilidad de brillo y color, basta con aplicar la normalización típica de modelos preentrenados (mean/std de ImageNet).
- **Augmentations moderados**: la homogeneidad del dataset sugiere utilizar aumentos suaves (flips, jitter ligero) para mejorar generalización sin distorsionar objetos.
- **Manejo de desbalance**: será necesario emplear *class weighting*, muestreo estratificado o pérdidas alternativas para evitar que el modelo favorezca categorías dominantes.
- **Selección del modelo**: la baja resolución original y la estructura limpia de producto sobre fondo blanco justifican el uso de *Vision Transformers*, que

aprovechan patrones espaciales uniformes y generalizan bien ante variaciones mínimas.

En conjunto, el EDA confirma que el dataset es adecuado para entrenar un sistema de clasificación de atributos visuales basado en arquitecturas modernas de visión profunda, requiriendo únicamente ajustes controlados para alcanzar condiciones óptimas de entrenamiento.

Capítulo 3

Preparación y Preprocesamiento

Este capítulo describe el proceso de preparación de datos realizado antes del entrenamiento del modelo. Se detalla cómo se construyeron los conjuntos de *train*, *validation* y *test*, los pasos de preprocesamiento aplicados a las imágenes y las técnicas de *data augmentation* utilizadas para mejorar la capacidad de generalización del modelo. Asimismo, se justifica la necesidad de normalizar y estandarizar las entradas de acuerdo con los requerimientos de la arquitectura *Vision Transformer* seleccionada.

3.1. División del dataset

A partir del conjunto limpio generado en la etapa de análisis exploratorio, se definió una división estratificada en tres subconjuntos: entrenamiento, validación y prueba. La estratificación se realizó tomando como variable objetivo el atributo *articleType*, con el fin de preservar la proporción relativa de clases en cada partición y evitar sesgos durante el entrenamiento.

El dataset final se dividió con los siguientes porcentajes:

- **70 %** para entrenamiento.
- **15 %** para validación.
- **15 %** para prueba.

La división se realizó mediante una estrategia robusta para manejar clases con baja frecuencia. Aquellos tipos de producto con menos de cinco muestras fueron agrupados en una clase adicional *RARE_CLASS* con el fin de evitar errores de estratificación. Esta técnica permitió mantener la coherencia estadística de los subconjuntos sin perder información relevante.

Además de los archivos `train.csv`, `val.csv` y `test.csv`, se generaron carpetas independientes con las imágenes correspondientes a cada partición, asegurando un pipeline reproducible y coherente con las buenas prácticas de *cookiecutter data science*:

```
data/
  processed/
    images/
      train/
      val/
      test/
```

Esta organización facilita la carga del dataset en PyTorch, reduce errores de lectura y estandariza el flujo de experimentación.

3.2. Preprocesamiento de imágenes

El preprocesamiento se orientó a adaptar las imágenes del dataset a los requerimientos del modelo *Vision Transformer*, cuya arquitectura espera entradas cuadradas y normalizadas. A partir del análisis realizado en el capítulo de EDA, se observaron las siguientes características relevantes:

- Todas las imágenes poseen tamaño uniforme de **60×80 píxeles**, inferior al requerido por ViT.
- La relación de aspecto es constante, lo que facilita el escalado sin distorsión.
- La calidad de las imágenes (luminosidad, contraste, nivel de enfoque y saturación) es adecuada para un modelo de clasificación supervisada, aunque con variabilidad suficiente como para beneficiar el uso de técnicas de *augmentation*.

En consecuencia, se aplicaron los siguientes pasos de preprocesamiento:

1. **Redimensionamiento a 224×224 píxeles.** Este tamaño es estándar para modelos basados en Transformers y permite aprovechar pesos preentrenados en ImageNet.
2. **Conversión a tensor y normalización.** Se aplicaron las estadísticas de ImageNet:
   $$\mu = (0{,}485, 0{,}456, 0{,}406), \quad \sigma = (0{,}229, 0{,}224, 0{,}225)$$
   con el fin de estabilizar el entrenamiento y lograr compatibilidad con pesos preentrenados.
3. **Validación de integridad.** Se eliminaron imágenes corruptas o no legibles y se depuraron filas del CSV que no podían vincularse a su archivo correspondiente.

Este pipeline garantiza que todas las imágenes de entrada cumplen el formato necesario para maximizar el desempeño del modelo.

3.3. Data augmentation

Dado que las imágenes del dataset son relativamente pequeñas y presentan poca diversidad visual en términos de iluminación, pose y fondo, se definió un conjunto de transformaciones destinadas a aumentar la variabilidad sintética del conjunto de entrenamiento. Esto reduce el riesgo de sobreajuste y mejora la robustez del modelo.

Las transformaciones aplicadas durante el entrenamiento fueron:

- **Random Horizontal Flip:** simula variaciones naturales en pose y orientación.
- **Color Jitter (brillo, contraste, saturación):** incrementa la tolerancia del modelo a condiciones de iluminación heterogéneas.

- **Random Rotation (±15°):** introduce pequeñas perturbaciones para mejorar la invariancia rotacional.
- **Affine transformations:** ligeras modificaciones de escala o traslación, cuando aplicable.

Estas transformaciones se aplican de manera estocástica únicamente al conjunto de entrenamiento. El conjunto de validación y prueba solo recibe las transformaciones estrictamente necesarias (redimensionamiento y normalización), con el objetivo de realizar una evaluación justa y reproducible del modelo.

3.4. Construcción del PyTorch Dataset

Con las particiones y el preprocesamiento definidos, se implementó una clase `PyTorch Dataset` capaz de:

- Cargar imágenes desde las carpetas correspondientes a cada split.
- Aplicar automáticamente las transformaciones definidas para cada partición.
- Mapear cada imagen hacia sus atributos objetivos codificados.
- Soportar clasificación multi-etiqueta mediante vectores binarios por atributo.

Este diseño permite integrar el dataset de manera directa con el *DataLoader* de PyTorch, facilitando el batching, el shuffle y la paralelización en CPU.

Apéndice A

Planificación del Proyecto

A continuación se presenta la tabla detallada de planificación del proyecto, que incluye todas las tareas necesarias para completar el trabajo desde la inicialización del repositorio hasta la presentación final.

TABLA A.1. Planificación del Proyecto

Título	Descripción	Responsable	Estado
Inicializar cookiecutter	Ejecutar `cookiecutter-data-science` para generar la estructura base del proyecto.	Abril	Completado
Configurar entorno Conda	Crear entorno `product_tagger` e instalar dependencias.	Abril	Completado
Crear repositorio GitHub	Crear repositorio, subir estructura inicial, conectar con VSCode.	Abril	Completado
Definir proyecto final	Acordar que el proyecto será un sistema Product Tagger basado en Vision Transformers.	Equipo	Completado
Definir Business Case	Documentar problema de negocio, costos y beneficios.	Abril	Completado
Obtención de Datos	Implementar descarga automática del dataset desde `KaggleHub`.	Abril	Completado
EDA	Explorar `styles.csv`, distribución de atributos y ejemplos de imágenes.	Abril	Completado
Train/Val/Test Split	Crear particiones estratificadas	Abril	Completado
Preparación imágenes	Redimensionar y normalizar imágenes para ViT.	Abril	Completado
Implementar PyTorch Dataset	Construir clase Dataset para imágenes y etiquetas.	XX	Pendiente
Implementar augmentations	Agregar normalización, resize, flips y transforms.	XX	Pendiente
Cargar Vision Transformer	Cargar ViT/DeiT preentrenado y adaptar classifier.	XX	Pendiente
Pipeline de entrenamiento	Entrenar ViT con optimizer, scheduler y early stopping.	XX	Pendiente
Registrar métricas	Guardar loss, accuracy, F1 y mAP por época.	XX	Pendiente
Evaluar modelo	Evaluar rendimiento sobre test set.	XX	Pendiente
Visualizar resultados	Graficar curvas, métricas y matriz de confusión.	XX	Pendiente
Ejemplos de predicción	Mostrar aciertos y errores.	XX	Pendiente
Guardar modelo entrenado	Exportar modelo final a `models/`.	XX	Pendiente
Escribir README	Instrucciones de instalación, entrenamiento e inferencia.	XX	Pendiente
Redactar Objetivo	Escribir objetivo del proyecto en el informe.	XX	Pendiente
Redactar Arquitectura	Diagramar y explicar el pipeline.	XX	Pendiente
Redactar Implementación técnica	Detallar módulos y diseño del sistema.	XX	Pendiente
Redactar Evaluación	Explicar métricas y análisis de resultados.	XX	Pendiente
Redactar Resultados	Incluir visualizaciones y análisis final.	XX	Pendiente
Redactar Conclusiones	Limitaciones y líneas futuras.	XX	Pendiente
Preparar presentación	Diapositivas y narrativa final.	XX	Pendiente
Practicar presentación	Ensayo del equipo.	XX	Pendiente
Entrega final código	Limpiar repo y hacer tagging final.	XX	Pendiente
Entrega final PDF	Compilar y entregar memoria final.	XX	Pendiente