Prueba técnica Departamento de datos no estructurados: Clasificación de imágenes

Autor: Nicolás Useche Narváez

Según Gartner, en este momento, los datos no estructurados conforman el 80-90% del total de los datos que manejan las compañías. Dentro de ellos, las imágenes de documentos ocupan un lugar importante. Cada año se hace más evidente la necesidad de convertir los datos contenidos en estas imágenes en información útil que pueda ser analizada en bases de datos.

El objetivo de este reto es lograr un filtro que discrimine automáticamente un tipo de documento sin información relevante: páginas en blanco. Se busca que este filtro reciba como entrada una carpeta con imágenes de documentos diversos y produzca como salida dos carpetas, una con imágenes de páginas sin contenido y otra con imágenes de páginas con contenido.

Páginas con solo el membrete del documento se consideran páginas sin contenido, así como las que, al momento de ser escaneadas, alcanzan a reflejar contenido ininteligible del reverso de la página.

1. Estrategia de Análisis

Para la solución de esta problemática, inicialmente se debe conocer que la necesidad es realizar una clasificación de imágenes de manera automatizada Teniendo en cuenta esto, se decide utilizar la técnica de transfer learning, permitiendo hacer uso de las primeras capas de la red neuronal convolucional VGG16 como extractor de características, las cuales serán el vector de entrada para el respectivo entrenamiento y clasificación que hará el modelo de randomForest, como se muestra en la figura 1.

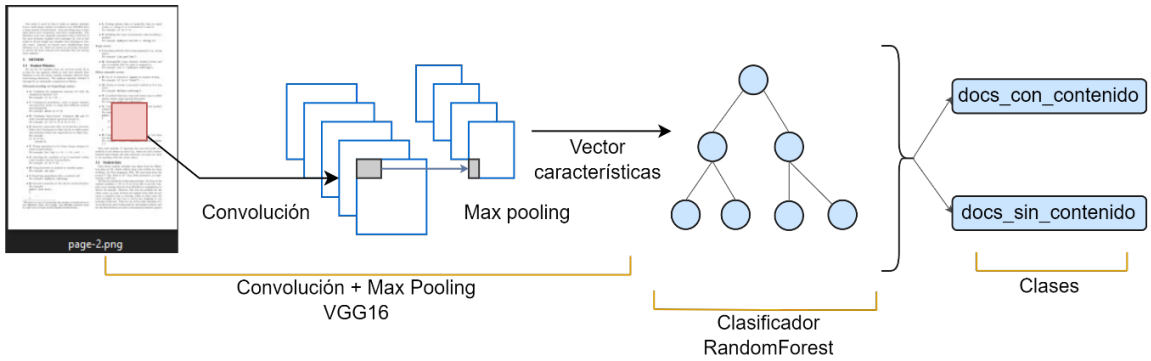


Figura 1. Arquitectura del modelo

2. Tecnologías y Librerías Utilizadas

Para el desarrollo del proyecto se empleó el lenguaje de programación Python en su versión 3.9.12 haciendo uso de librerías como “numpy” para el manejo de matrices. “seaborn” para ilustraciones y gráficos. Para la lectura y preprocesamiento de imágenes se empleó la librería “opencv”, “os” y “glob” para el manejo de archivos y directorios. Además, se utilizó “sklearn” para el modelo de clasificación Random Forest, el etiquetado de las imágenes y las métricas del modelo. “keras” para importar la red neuronal convolucional VGG16 preentrenada para la abstracción de características.

3. Exploración de Datos

En la exploración del dataset se identificaron 247 imágenes en formato png, las cuales estaban divididas en dos clases: “docs_con_contenido” y “docs_sin_contenido” con 100 y 147 imágenes respectivamente. Lo anterior puede significar que hay un desbalance en los datos, pero en el desarrollo de esta prueba no significó un problema.

Finalmente se realiza una inspección visual de las imágenes del dataset, a partir de la cual se tienen las siguientes observaciones:

- Se encontró que las imágenes no tenían las mismas dimensiones (ancho y alto).
- Las imágenes de los “docs_sin_contenido” presentan características distintivas como: letras en el reverso de la página, páginas grises, membretes, anotaciones manuales, etc. Igualmente se puede observar que todas son imágenes de documentos escaneados.
- Las imágenes de “docs_con_contenido” también son documentos escaneados, que presentan texto agrupado en diferentes ubicaciones de la página. En algunos casos se presentan portadas, gráficos y tablas de contenido.

4. Preparación de los datos

Paso Previo: División de conjunto de datos

Esta división se realizó de manera manual conservando la siguiente proporción:

- **train (80%)**: 80 imágenes de “docs_con_contenido” y 118 imágenes “docs_sin_contenido”.
- **validation (20%)**: 20 imágenes de “docs_con_contenido” y 29 imágenes de “docs_sin_contenido”.

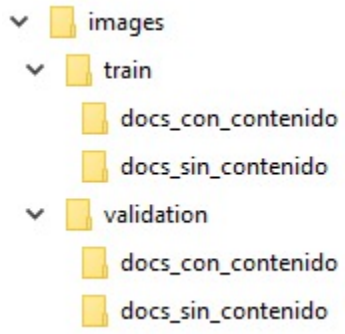


Figura 2. Estructura de archivos.

Este paso permite la facilidad del etiquetado de los datos de manera automática como se presenta en el paso 3 del código desarrollado para la solución de la prueba.

Teniendo el dataset debidamente estructurado, se realiza el preprocesamiento de los datos que consiste en:

- Importación del dataset.
- Acondicionamiento de la imagen, incluyendo operaciones como resize a 256x256 pixeles y transformaciones de espacio de color.
- Normalización de la imagen a 8bit.
- Etiquetado de cada imagen según su categoría ("docs_con_contenido" o "docs_sin_contenido"), esta etiqueta se toma del nombre de la carpeta donde se encuentra cada imagen de train y validation, para luego ser codificada como "0" y "1".

5. Extracción de características

Se aplican técnicas de extracción de características para mejorar el rendimiento del modelo, teniendo en cuenta que permiten reducir de manera eficiente el tamaño de los datos, al extraer las mejores características posibles de las imágenes (como por ejemplo colores, bordes o formas). En este caso se hace uso de las primeras capas del modelo VGG16 el cual es una red neuronal convolucional con 16 capas de profundidad. Esta red neuronal está preentrenada con más de un millón de imágenes de la base de datos de ImageNet, que ya tiene sus pesos establecidos. [1]

6. Selección y entrenamiento del modelo

Se selecciona el modelo de Random Forest, teniendo en cuenta que hace parte de las técnicas de árbol de decisión, las cuales se consideran uno de los métodos más conocidos para la clasificación de datos [2]. Además, el Random Forest tiende a ser menos propenso al sobreajuste en comparación con otros modelos más complejos, como las redes neuronales profundas, especialmente cuando se cuenta con conjuntos de datos relativamente pequeños [3], como en este caso que el dataset solo tiene 247 imágenes.

7. Evaluación del modelo

Para evaluar el rendimiento del modelo de Random Forest en la tarea de clasificación de imágenes, se utilizó la métrica de exactitud (accuracy). La cual es una medida fundamental que indica la proporción de predicciones correctas realizadas por el modelo con respecto al total de predicciones realizadas.

Se compararon las predicciones realizadas por el modelo con las etiquetas verdaderas de las imágenes en el conjunto de prueba haciendo uso de una matriz de confusión, como se observa en la siguiente figura.

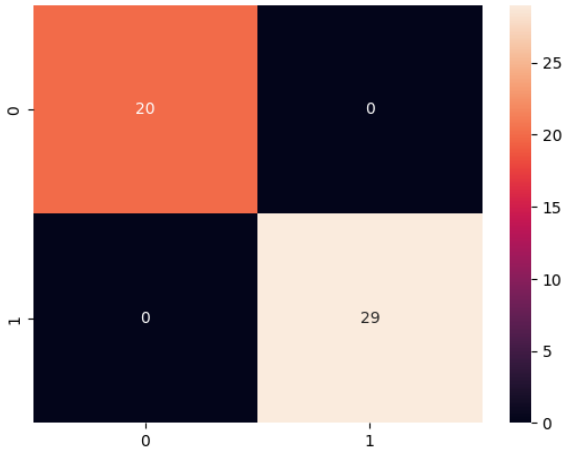


Figura 3. Matriz de confusión

8. Aplicación del filtro

En esta sección se implementa una función auxiliar que utiliza el modelo previamente entrenado para realizar la clasificación de una imagen llamada predict_image. Adicionalmente, se agrega la función que filtra las imágenes según el requerimiento, este filtro recibe como entrada la carpeta "total" creada manualmente (contiene las 247 imágenes correspondientes a las clases de "docs_sin_contenido" y "docs_con_contenido") y produce como salida dos carpetas, una con imágenes de páginas sin contenido y otra con imágenes de páginas con contenido.

9. Interpretación de Resultados

Se realiza la matriz de confusión como se muestra en el paso 6, teniendo en cuenta las imágenes de test que se utilizaron en el entrenamiento del modelo, que corresponden al 20% del dataset (49 imágenes).

La matriz de confusión de la figura 3, indica que el modelo tiene un rendimiento perfecto al distinguir imágenes de documentos con contenido y sin contenido, es decir no presenta ningún error en la clasificación.

Si analizamos cada valor obtenido en la matriz se tiene que:

- Verdaderos positivos (TP): Hay 20 imágenes clasificadas correctamente como documentos con contenido.
- Falsos negativos (FN): El valor es 0, lo que muestra que no se clasificaron documentos con contenido como sin contenido.
- Falsos positivos (FP): Se tiene un valor de 0. Esto indica que no se clasificaron documentos sin contenido como con contenido. No tener falsos positivos es bueno, ya que significa que no está etiquetando erróneamente imágenes de documentos en blanco o irrelevantes como documentos con contenido.
- Verdaderos negativos (TN): Hay 29 imágenes clasificados correctamente como documentos sin contenido.

La medida escogida para evaluar el modelo es la exactitud **accuracy** se calcula como la suma de los verdaderos positivos y los verdaderos negativos dividida por el total de la muestra.

En este caso, sería

$$\frac{20+29}{20+29+0+0} = \frac{49}{49} = 1$$

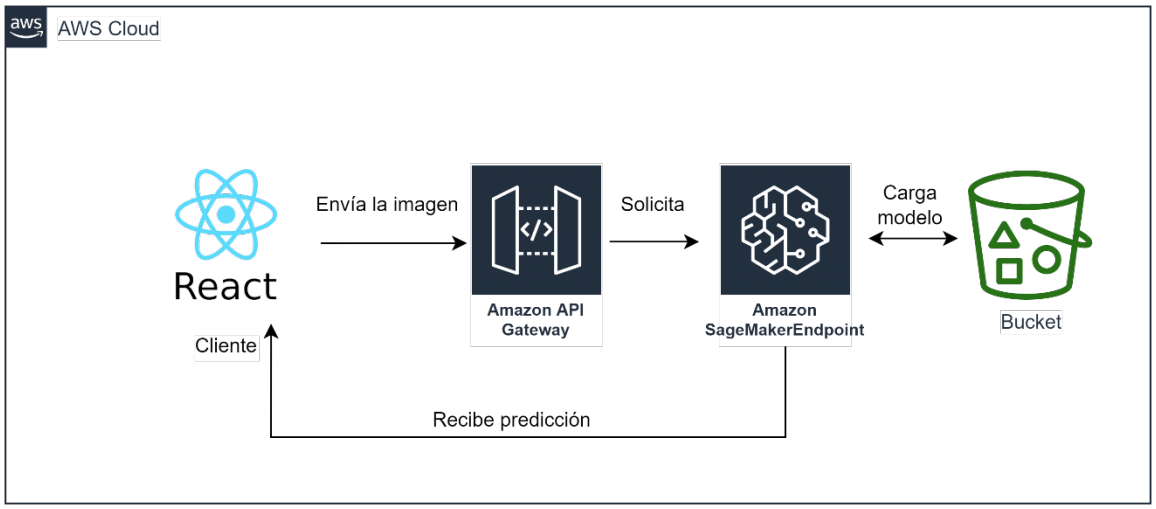
Esto sugiere que el modelo tiene una tasa de aciertos del 100%, lo cual es excelente.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

10. Despliegue

Para la puesta en producción del modelo de clasificación de imágenes desarrollado previamente, se puede hacer a través de la nube de AWS, haciendo uso de servicios como AWS S3, SageMaker, API Gateway y CloudFront.

Para esto se exporta el modelo VGG16, Random Forest y el label_encoder(decodificar el resultado de la predicción) en un archivo .pkl y se sube al bocket S3 de AWS para luego ser usado por Sagemaker.



- Cliente: La aplicación cliente que consume el servicio de clasificación de imágenes. Puede ser una aplicación web, móvil, o cualquier otro sistema capaz de enviar solicitudes HTTP.
- API Gateway: Servicio de AWS que actúa como una puerta de enlace para las APIs, gestionando las solicitudes entre el cliente y los servicios backend como SageMaker.
- SageMaker Endpoint: Endpoint creado por Amazon SageMaker que aloja el modelo de clasificación de imágenes y procesa las peticiones entrantes, realizando las predicciones.
- Modelo en S3: El modelo de clasificación de imágenes almacenado en un bucket de AWS S3, desde donde SageMaker lo carga para realizar inferencias.

Este flujo garantiza un proceso escalable y seguro para poner en producción el modelo de clasificación de imágenes utilizando AWS.

11. Referencias

1. https://la.mathworks.com/help/deeplearning/ref/vgg16.html

2. Laliberte, A. S., Koppa, J., Fredrickson, E. L., & Rango, A. (2006, July). Comparison of nearest neighbor and rule-based decision tree classification in an object-oriented environment. In Proceedings of IEEE international Geoscience and Remote Sensing Symposium, Denver, Colorado, USA (pp. 3923-3926).

3. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32. https://doi.org/10.1023/A:1010933404324