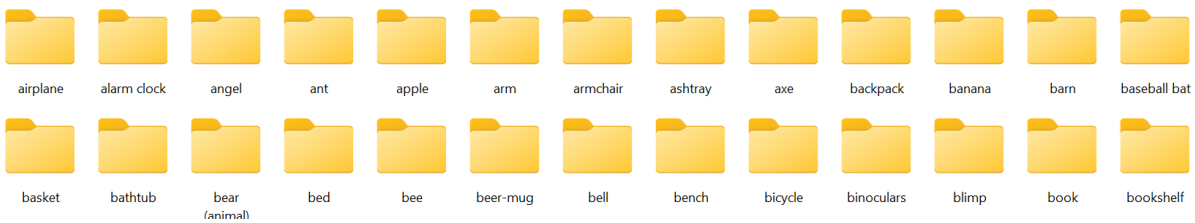
Práctica 2

1. Enunciado

Esta práctica está basada en un artículo presentado en SIGGRAPH, una de las conferencias más importantes (si no es la que más) en informática gráfica. El artículo presentaba un estudio a gran escala donde se analizaba la distribución de los bocetos de objetos cotidianos como "tetera" o "coche" realizados por personas no expertas. En concreto, se reunieron 20.000 dibujos únicos distribuidos uniformemente en 250 categorías de objetos. En la figura de abajo se muestra un ejemplo de algunos de estos bocetos.

Ejemplos de dibujos en el conjunto de datos

En `http://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/` se puede encontrar toda la información al respecto, así como los archivos SVG de cada boceto y su rasterizado en PNG (que es 10 veces más pesado, claro). En la imagen de abajo se muestran solo las 26 primeras clases, que son las que utilizaremos para esta práctica. El conjunto de datos tiene un tamaño de 50 MB aproximadamente.



Carpetas con las 26 primeras clases

El proyecto consiste en crear un sistema para clasificar bocetos.
Para ello se deben realizar los siguientes scripts:

1. `splitter.py`

   Debe crear un script que genere 3 ficheros de texto:

   `train.txt` contendrá la ruta y nombre de aquellos ficheros que utilizaremos para entrenar y se utilizará **sólo** en el script `train.py`. Debe contener el 80 % de los ejemplos y la separación debe ser estratificada.

   `validation.txt` contendrá la ruta y nombre de aquellos ficheros que utilizaremos para validar el modelo y se utilizará **sólo** en el script `train.py`.
   Debe contener la mitad de los ejemplos NO elegidos en `train.txt` y la separación debe ser estratificada.

`test.txt` contendrá la ruta y nombre de aquellos ficheros que utilizaremos para simular los nuevos ejemplos que puedan llegar cuando el modelo ya está en producción, y se utilizará **sólo** en el script `finalproduct.py`.
Debe contener la **otra** mitad de los ejemplos no elegidos en `train.txt` y la separación debe ser estratificada.

2. `train.py`

   Debe leer **sólo** las imágenes de `train.txt` y `validation.txt`. El primero sólo se debe usar para entrenar un modelo de clasificación, mientras que el segundo se puede utilizar cada evaluar el proceso de entrenamiento. Al terminar el entrenamiento se debe guardar el modelo en un fichero.

3. `finalproduct.py`

   Debe leer **sólo** las imágenes de `test.txt` y también debe cargar el modelo aprendido. Después debe pasar cada una de las imágenes de test al modelo y obtener una estimación de su etiqueta. Al terminar debe generar un fichero de texto llamado `etiquetas_estimadas.txt` y otro donde se impriman resultados de la matriz de confusión llamado `matriz_confusion.txt`

2. Condiciones de entrega

- Se realiza 1 práctica por grupo y se mantendrán los mismos grupos que en la práctica anterior. En caso de NO querer cambiar de grupo se debe notificar al profesor.
- Se puede utilizar el código proporcionado en clase; y si se utiliza código de terceros debe estar indicado con el comentario `#** codigo de terceros ! **`
- La fecha límite para subir el fichero ZIP aparece en la entrega del aula virtual. Este fichero ZIP debe contener:
  - un fichero **nombres.txt** con el nombre de los alumnos del grupo
  - los scripts `splitter.py`, `train.py` y `finalproduct.py`
  - los ficheros de texto `train.txt`, `validation.txt`, `test.txt`, `etiquetas_estimadas.txt` y `matriz_confusion.txt`
  - una breve **memoria** (no más de 4 páginas + portada) explicando como se ha abordado el problema, qué redes se han construido para ello y qué parámetros de entrenamiento se han empleado.
- **NO** se debe subir el conjunto de datos dado ni tampoco el modelo.
  El límite de subida es de 5 MB precisamente para evitar esto.