Tarea 1

CC5213 – Recuperación de Información Multimedia
Profesor: Juan Manuel Barrios
Fecha de entrega: 28 de abril de 2025

El objetivo de esta tarea es implementar un **buscador de imágenes duplicadas o derivadas**. Dado un conjunto de imágenes originales **R** y un conjunto de imágenes derivadas **Q**, se desea determinar para cada imagen *q* ϵ **Q** si existe alguna imagen *r* ϵ **R** tal que *T(r)=q* para alguna transformación *T*.

Figura 1: Ejemplo del conjunto **R** con imágenes originales

Figura 2: Ejemplos del conjunto **Q** con imágenes derivadas

Llamaremos *transformación* a una función *T* que recibe como entrada una imagen *r* y genera como salida una imagen *q* que es una modificación de *r.* En la Figura 1 se ven imágenes originales obtenidas del dataset **mirflickr25k**[1] y en la Figura 2 se ven imágenes derivadas correspondientes. En la tarea se evaluarán las siguientes transformaciones: bajar calidad, insertar texto, ajustar colores, recortar una zona (crop), reflejado horizontal (flip), perspectiva (homografía), agregar bordes.

[1] MIRFLICKR: https://press.liacs.nl/mirflickr/

Para la tarea debe implementar dos comandos:

1. **Fase offline del sistema**: Un comando que recibe el nombre de carpeta donde están las imágenes originales **R** y el nombre de una carpeta nueva donde se guardarán sus datos:

   `python tarea1-parte1.py` *`dir_imagenes_R dir_descriptores_R`*

   Para cada imagen en `dir_imagenes_R` calcula un descriptor de contenido (a su elección). Al finalizar guarda en la carpeta `dir_descriptores_R` uno o más archivos con los nombres de las imágenes procesadas y sus descriptores.

2. **Fase online del sistema**: Un comando que recibe el nombre de la carpeta donde están las imágenes derivadas **Q**, la carpeta con los descriptores de **R** (creada por la fase offline) y el nombre del archivo de salida a crear con los resultados:

`python tarea1-parte2.py` *`dir_imagenes_Q dir_descriptores_R resultados.txt`*

   Para cada imagen de **Q** calcula un descriptor de contenido, que podría no ser necesariamente el mismo método usado para **R** (a su elección). Luego para cada descriptor de **Q** se busca el descriptor más cercano de **R**. Finalmente, genera un archivo de texto con el resultado de imágenes más parecidas.

   El archivo de texto a generar debe tener una fila por imagen en **Q** y en cada fila tres columnas separadas por tabuladores: nombre de imagen de **Q**, nombre de imagen de **R** más parecida y la distancia entre ambas:

   `nombre_imagen_q` ***`\t`*** `nombre_imagen_r` ***`\t`*** *`distancia_entre_q_r`*

   Notar que mientras más parecidas sean las imágenes, menor debe ser la distancia entre sus descriptores.

Datos de prueba

Junto con este enunciado encontrará una implementación base para `tarea1-parte.py` y `tarea1-parte2.py`, tres conjuntos de prueba con imágenes originales y transformadas (llamados `dataset_a`, `dataset_b`, `dataset_c`), y un programa de evaluación con la respuesta esperada para cada imagen de consulta.

Evaluación

Antes de entregar su tarea, debe utilizar el programa de evaluación. Para usar el programa de evaluación debe invocar el programa:

```
python evaluarTarea1.py
```

Este programa llama a `tarea1-parte1.py` y `tarea1-parte2.py` con cada dataset de prueba, lee los resultados generados, compara las respuestas con las respuestas correctas en cada dataset, determina la distancia umbral de corte que logra un mejor balance entre respuestas correctas e incorrectas, y finalmente dependiendo de la cantidad de detecciones correctas reporta la nota obtenida.

Su tarea será evaluada en los conjuntos de imágenes publicados y en otros conjuntos similares. Su tarea **no puede demorar más de 15 minutos** en procesar cada conjunto de prueba.

Existe la posibilidad de obtener **hasta 1 punto extra de bonus** para otras tareas si logra detectar la mayoría de los duplicados en los datasets de evaluación.

Entrega

El plazo máximo de entrega es el **lunes 28 de abril de 2025** hasta las 23:59 por U-Cursos. Existirá una segunda fecha de entrega sin descuentos, por definir.

La tarea la puede implementar en **Python 3** usando cualquier función de OpenCV, NumPy, SciPy y otras librerías gratuitas (exceptuando librerías de machine learning). Debe subir sólo el código fuente de su tarea (archivos .py). No envíe datasets ni descriptores ya calculados.

Se recomienda incluir un archivo de texto señalando el sistema operativo en que realizó su tarea e incluir la salida que entregó `evaluarTarea1.py` en su computador.

Opcionalmente, puede implementar la tarea en **C++ 17**. En este caso puede usar cualquier función de OpenCV y de la biblioteca estándar (std). Debe subir el código fuente de su tarea junto con los pasos necesarios para la compilación.

La tarea es *individual* y debe ser de su autoría, es decir, no pueden ser resueltos por otro estudiante, no se pueden copiar respuestas de Internet, no se permite usar ChatGPT ni similares. En caso de detectar copia o plagio se asignará nota 1.0 a las o los estudiantes involucrados.