Tarea 1

CC5213 – Recuperación de Información Multimedia
Profesor: Juan Manuel Barrios
Fecha: 7 de septiembre de 2020

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

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 edición de r. Una transformación será válida cuando es posible reconocer el contenido de r en q.

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, recortar una zona, insertar texto, ajustar colores y una combinación de todas ellas.

La tarea se debe dividir en dos módulos:

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

   ```
   python tarea1-procesar.py [dir_imágenes_R] [datos_R]
   ```

   Para cada imagen en `dir_imágenes_R` se calcula su descriptor de contenido y al finalizar guarda en la carpeta `datos_R` uno o más archivos con los nombres de las imágenes procesadas y sus descriptores.

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

2. Un comando que recibe el nombre de la carpeta donde están las imágenes a buscar **Q**, la carpeta con los datos de **R** (creada por el comando anterior) y el nombre del archivo de resultados a crear:

```
python tarea1-buscar.py [dir_imágenes_Q] [datos_R] [resultados.txt]
```

   Se cargan los vectores en `datos_R`, se calculan los vectores de las imágenes en `dir_imágenes_Q`, para cada vector de **Q** se obtiene el vector más cercano de **R** y finalmente genera un archivo de texto con el resultado. El archivo de texto debe tener una fila por imagen en **Q** y para cada fila tres columnas separadas por tabuladores con el resultado de la búsqueda:

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

   Notar que mientras más similares sean las imágenes, menor es la distancia entre sus descriptores.

Junto con este enunciado encontrará dos conjuntos de imágenes de prueba y un programa `tarea1-evaluar.py` que ejecuta su tarea y evalúa el resultado generado.

Deberá entregar los códigos fuentes de su tarea y un archivo `Readme.txt` que explica brevemente su implementación, el o los descriptores implementados, librerías usadas, forma de compilación y resultados obtenidos.

Su tarea será evaluada en los conjuntos de imágenes publicados y en otros conjuntos similares. No debe enviar imágenes ni archivos con descriptores. Su tarea no puede demorar más de 1 hora en procesar cada conjunto de imágenes de prueba publicado.

Entrega:

- El plazo máximo de entrega es el **lunes 28 de septiembre** hasta las 23:59 por U-Cursos. No se aceptarán tareas atrasadas.
- La tarea la puede implementar en **Python 3**, **C++ 11** o **Java 8** usando la librería **OpenCV**. Puede utilizar cualquier función de OpenCV y SciPy.
- Enviar solo el código fuente de su tarea (archivos .cpp, .py o .java) y el archivo `Readme.txt`. No enviar imágenes ni descriptores ya calculados.

Evaluación:

- Se evaluarán tres aspectos: orden del código fuente (20%), claridad del archivo `Readme.txt` (20%) y la calidad de los resultados logrados (60%).
- La calidad de los resultados logrados se medirá ejecutando el comando `tarea1-evaluar.py` sobre distintos conjuntos de imágenes.
- La tarea es ***individual***.