

Proyecto Final Área Imágenes - CBIR

Búsqueda de Imágenes Similares Basado en su Contenido

Integrantes: Benjamín Irarrázabal
Joaquín Zepeda
Profesor: Claudio Pérez
Auxiliar: Juan Pablo Pérez
Ayudantes: Jorge Zambrano
Ayudante de laboratorio: Diego Maureira

Fecha de realización: 14 de julio de 2022
Fecha de entrega: 14 de julio de 2022
Santiago de Chile

Índice de Contenidos

| | |
|--|-----------|
| 1. Introducción | 1 |
| 2. Métodos empleados | 2 |
| 3. Medida de similitud | 4 |
| 4. Ranking | 5 |
| 4.1. INRIA Holydays dataset Rank | 5 |
| 4.2. GPR1200 Rank | 6 |
| 5. Algoritmo de búsqueda | 7 |
| 6. Resultados | 8 |
| 6.1. INRIA Holidays database | 8 |
| 6.2. GPR1200 | 13 |
| 7. Análisis de resultados | 18 |
| 7.1. Análisis Cualitativo | 19 |
| 7.2. Análisis Cuantitativo | 19 |
| 8. Optimización | 19 |
| 9. Conclusión | 21 |
| Referencias | 22 |
| 10. Anexos | 23 |
| 10.1. Repositorio del Curso | 23 |
| 10.2. Instrucciones Proyecto | 23 |

Índice de Figuras

| | |
|---|----|
| 1. Ejemplos de los métodos Handcrafted. | 2 |
| 2. | 3 |
| 3. Imagen referencial de la arquitectura de la red VGG16. | 3 |
| 4. Gráfico del vector de características entregado por la red VGG16 con la imagen del ejemplo de la figura 1. | 4 |
| 5. Gráfico comparativo entre el ranking obtenido para ambos métodos de extracción. . . . | 6 |
| 6. Gráfico comparativo entre el ranking obtenido para ambos métodos de extracción . . . | 7 |
| 7. Imagen consultada para ambos extractores | 8 |
| 8. Resultados usando el método clásico | 9 |
| 9. Resultados usando el método CNN | 10 |
| 10. Imagen consultada para ambos extractores | 11 |
| 11. Resultados usando el método clásico | 11 |

| | | |
|-----|--|----|
| 12. | Resultados usando el método CNN | 12 |
| 13. | Imagen de consulta para ambos extractores | 13 |
| 14. | Resultados obtenidos para el método clásico | 14 |
| 15. | Resultados obtenidos para el extractor CNN | 15 |
| 16. | Imagen de consulta para ambos extractores | 16 |
| 17. | Resultados obtenidos para el extractor clásico | 17 |
| 18. | Resultados obtenidos para el extractor CNN | 18 |

Índice de Tablas

| | | |
|----|--|----|
| 1. | Resultados de la comparación de extractores según la medida de similitud para INRIA Holidays dataset | 6 |
| 2. | | 7 |
| 3. | Tiempo promedio y desviación en determinar la distancia euclíadiana entre 2 vectores de largo 25088 para el caso normal y el caso optimizado (tomando 100 muestras de tiempo). | 20 |

1. Introducción

Dentro de la Inteligencia Computacional, el Procesamiento Digital de Imágenes es una herramienta bastante útil. Esta área está compuesta por un conjunto de técnicas aplicadas a imágenes digitales, que permiten mejorar su calidad o facilitar la extracción de información de estas. Entre estas aplicaciones, nace la Búsqueda de Imágenes Similares Basado en su Contenido (*Content Based Image Retrieval*, CBIR).

En cada momento, miles de imágenes son subidas a internet, haciendo referencia a cierto tipo de contenido en particular, como por ejemplo, marcas, paisajes, entre otros. Junto a esto, las personas utilizan algunos buscadores (como por ejemplo Google), para encontrar imágenes similares de acuerdo a la consulta entregada. En este punto, CBIR se transforma en una herramienta fundamental.

Tal y como dice su nombre, CBIR permite buscar imágenes según el contenido de estas, es decir, usando algún extractor de características definido, se analizan sus colores, figuras, texturas u otro tipo de información contenida y la compara usando una medida de similitud (distancia) con las demás imágenes de la base de datos.

En base a esto, el siguiente trabajo tiene como objetivo la implementación de un algoritmo búsqueda de imágenes similares basado en su contenido utilizando dos tipos de extracción de características, uno clásico (Histograma) y una red convolucional pre-entrenada (VGG16). Con esto, el usuario podrá consultar por una imagen de la base de datos y el algoritmo usando una medida de similitud le entregará los 10 resultados más cercanos. Este algoritmo será implementado para dos bases de datos distintas, la primera correspondiente a INRIA Holidays dataset y la segunda a GPR1200.

Para esto, se comenzará introduciendo los extractores usados, junto a una breve descripción de su funcionamiento, para luego dar paso a la aplicación del algoritmo en ambas bases de datos utilizadas. Posteriormente, se entregarán los resultados más relevantes, comparando ambos casos (Clásico v/s CNN) y comentando sobre estos. Finalmente, en la sección de conclusiones, se destacarán los puntos más importantes de la experiencia, comentando además, las dificultades al realizar el proyecto.

Además, al final de este informe se presenta una sección de anexo (10) que contiene un Repositorio de GitHub con los proyectos de todo el semestre, junto al proyecto final, con sus archivos más importantes. También, se detallan las instrucciones para el manejo de los Notebooks entregados con este informe.

2. Métodos empleados

En primera instancia se utilizan extractores con el fin de extraer vectores de características para las imágenes. Se utilizan métodos Handcrafted los cuales usan las técnicas clásicas de representación de imágenes. Para definir los extractores se utilizó la librería de *scikit-image* la cual tiene implementado varios de estos métodos.

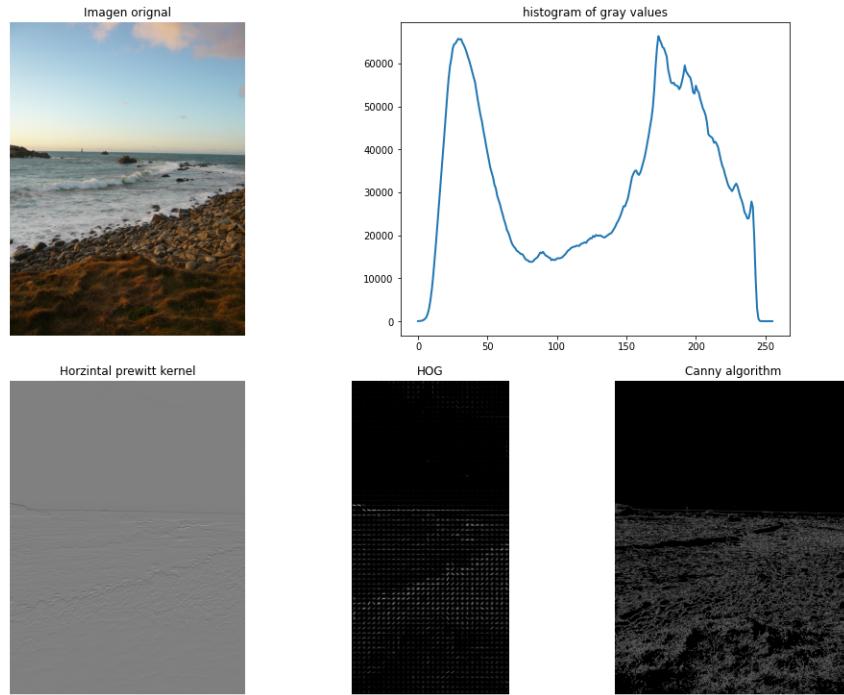


Figura 1: Ejemplos de los métodos Handcrafted.

Luego de explorar distintas técnicas se decidió utilizar el método del histograma RGB como características por su simplicidad. Este método dice cuantos píxeles de la imagen tienen determinado valor, de esta manera representa la imagen con un vector de largo 256. Es importante notar que se determina el histograma para los 3 canales de la imagen y luego se combinan en un solo histograma generado por la suma de estos 3. En la imagen 2 se puede observar un ejemplo del proceso de extracción.

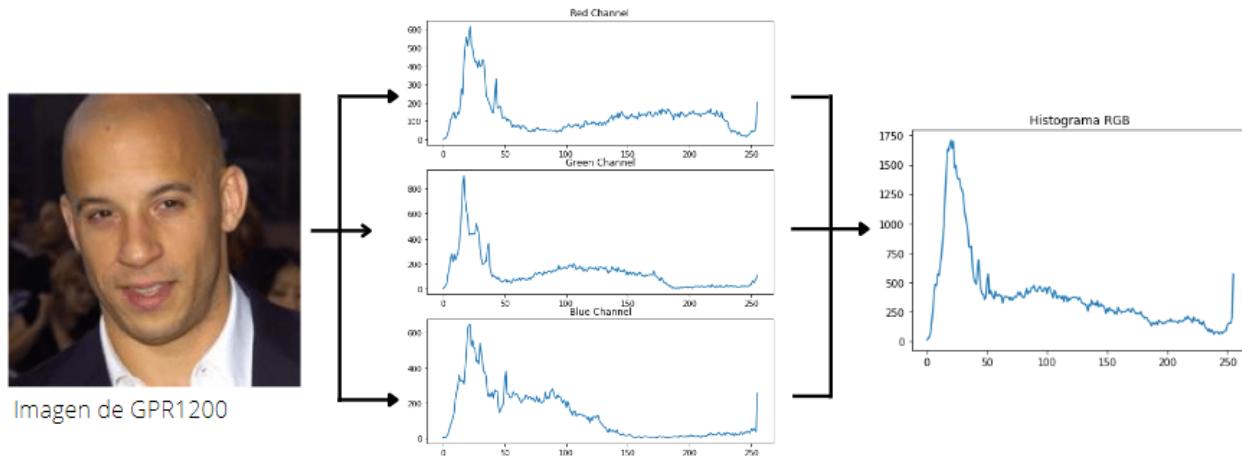


Figura 2

Por otro lado, también se definen extractores de características utilizando redes neuronales convolucionales (CNN) pre-entrenadas, se utilizó una red VGG16 pre-entrenada con la base de datos de **imagenet** (es decir utiliza los pesos encontrados al entrenar la red con esa base de datos) la cual corresponde a una gran base de datos visual diseñada para reconocer objetos visuales. Además se utilizan imágenes con *input_shape* = (224, 224, 3), por lo que a las imágenes se les realiza un *resize()* y un *reshape()* de la imagen para que tengan el tamaño adecuado para la red.

La arquitectura de la red VGG16 pre-entrenada se presenta en la imagen 3.

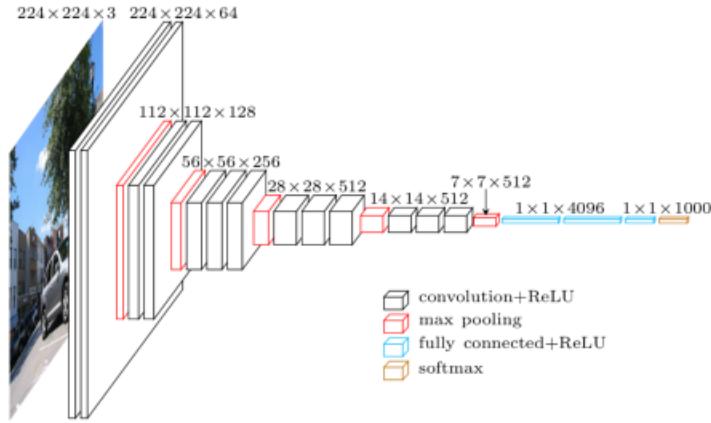


Figura 3: Imagen referencial de la arquitectura de la red VGG16.

Como se puede observar en la figura 3, esta es una red que tiene capas convolucionales y luego tiene capas fully connected, como se busca utilizar la red pre-entrenada para extraer las características, se desconectan las capas fully connected (con el parámetro *include_top=False*), dejando solo las capas convolucionales después del max pooling, esto se debe a que estas capas son las que extraen las características de la imagen que es lo que se busca utilizar. Esto se define de la siguiente manera:

```

1 # Se carga la red pre-entrenada
2 model = VGG16(include_top=False,
3     weights='imagenet',input_shape=( 224, 224, 3))

```

```

4 # remove the output layer
5 model = Model(inputs=model.inputs,outputs=model.layers[-1].output)
6
7 # get extracted features
8 features = model.predict(img)
9
10 # 1x7x7x512 = 25088
11 features.flatten()

```

Luego, se utiliza el método predict() del modelo para extraer el vector de 7x7x512 correspondiente a la salida de las redes convolucionales luego del max pooling, este vector se aplana utilizando el método flatten() para encontrar un vector unidimensional con las características extraídas, este vector tendrá un largo de 25088 luego de aplanarse.

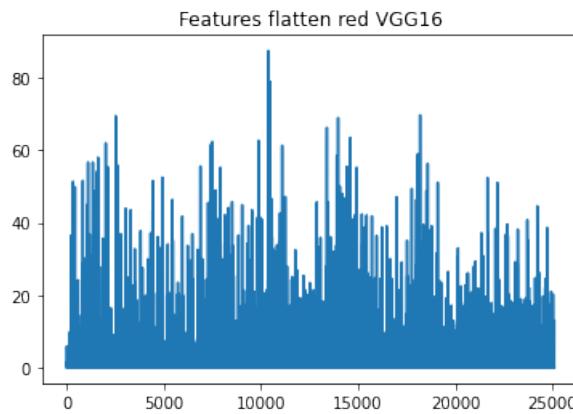


Figura 4: Gráfico del vector de características entregado por la red VGG16 con la imagen del ejemplo de la figura 1.

3. Medida de similitud

La medida de similitud definida corresponde a la distancia euclíadiana, la cual se define de la siguiente manera:

$$d(x, y) = \sqrt{\sum_{i=1}^n (y_i - x_i)^2} \quad (1)$$

Inicialmente se define una función propia que determina esta distancia recorriendo componente a componente, lo cual se cambia luego para optimizar el tiempo de computo de estos valores. En la sección de optimización se habla a detalle sobre este cambio.

```

1 from math import sqrt
2 def distancia_eucliana(x,y):
3     """
4         Medida de similitud entre vectores de características,
5         usando la distancia euclíadiana.
6

```

```

7  :param numpy.ndarray x: primer vector
8  :param numpy.ndarray y: segundo vector
9
10 :return: la distancia euclíadiana entre los vectores
11 """
12 suma = 0
13 for xi,yi in zip(x,y):
14     suma+= abs(xi-yi)**2
15 return sqrt(suma)

```

4. Ranking

Para comparar el desempeño de ambos extractores, se procedió a implementar una medida de Ranking Normalizado, el cual se rige por la siguiente ecuación:

$$Rank = \frac{1}{N \times N_{rel}} \left(\sum_{i=1}^{N_{rel}} R_i - \frac{N_{rel}(N_{rel} + 1)}{2} \right) \quad (2)$$

Con N_{rel} correspondiente al número de imágenes relevantes para una imagen de consulta particular, R_i la posición de las imágenes de la misma clase que la consultada y N el tamaño del set de imágenes. A continuación se detallan los pasos que realiza el algoritmo para aplicar Ranking, esto además se puede corroborar en los códigos adjuntos a este informe.

- Se definen variables o arreglos que guardarán resultados importantes.
- Se buscan las características de la imagen consultada en la base de datos.
- Se calcula el vector de distancias utilizando la medida de similitud especificada en la sección anterior.
- Se guardan las posiciones de las imágenes que presentan las mínimas distancias.
- Se calcula la cantidad de imágenes relevantes (N_{rel}) y se calcula el Ranking.
- Finalmente, la función presenta un parámetro que según sea 'Si' o 'No', devuelve una lista con los nombres de las 10 imágenes según ranking.

4.1. INRIA Holydays dataset Rank

Para esta base de datos se consideró $N = 991$ (cantidad de imágenes en dataset), luego, usando la función detallada anteriormente, se calculó el Ranking para todas las imágenes de consulta, utilizando ambos algoritmos.

Posteriormente, se realizó una comparación cuantitativa para ver cual de los dos métodos presenta un mejor ranking, considerando que para una misma imagen de consulta, el método con menor rank es más robusto. Los resultados de esta comparación se pueden observar en la siguiente tabla.

Tabla 1: Resultados de la comparación de extractores según la medida de similitud para INRIA Holidays dataset

| | |
|-----------------------|-----|
| Histogram_wins | 134 |
| CNN_wins | 325 |
| Empates | 41 |

Como se puede observar, para esta base de datos, la extracción de características y la posterior clasificación (ranking según similitud) es mejor utilizando la red convolucional VGG16 preentrenada. Sumado a esto, aplicar la función de rank normalizado a la base de datos completa tarda entre 20 y 30 segundos para el extractor clásico y entre 30 y 40 segundos para el extractor CNN.

A continuación, se presenta un análisis cualitativo a estos rankings, donde se representa en color azul, el valor obtenido para el método HandCrafted, mientras que en rojo se representa el ranking del método CNN.

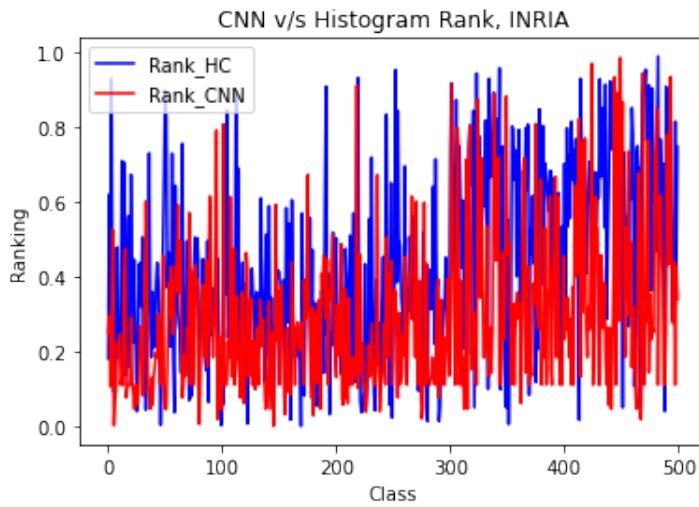


Figura 5: Gráfico comparativo entre el ranking obtenido para ambos métodos de extracción.

El gráfico de la figura 5 muestra claramente que los resultados para la red VGG16 son mejores, debido a que son menores con respecto a los del extractor clásico.

4.2. GPR1200 Rank

En esta segunda base de datos se considera un $N = 11999$, debido a que cuando se consulta por una imagen, esta se debe comparar con las 11999 restantes (sin contar la consultada para evitar la repetición de esta), además, se destaca que en este caso $N_{rel} = 9$, debido a que siempre, al consultar por una imagen, quedarán 9 más de la misma clase en la base de datos (ya que GPR1200 está balanceada con 10 ejemplos de cada clase). Con esto, se calculó el rank normalizado para ambos extractores. Cabe destacar, que como esta base de datos es mucho más grande, el algoritmo para encontrar los rankings de todas las imágenes tarda mucho más. En primer lugar, el ranking para el método clásico tardó 1 hora con 5 minutos aproximadamente, luego, el ranking para el método CNN. Análogo a la sección anterior, se muestran algunos resultados cuantitativos para este ranking.

Tabla 2

| | |
|-----------------------|------|
| Histogram_wins | 5437 |
| CNN_wins | 6562 |
| Empates | 1 |

En la tabla anterior, se puede observar que nuevamente el extractor CNN posee un mejor desempeño. A continuación, se muestra el análisis cualitativo de este Rank.

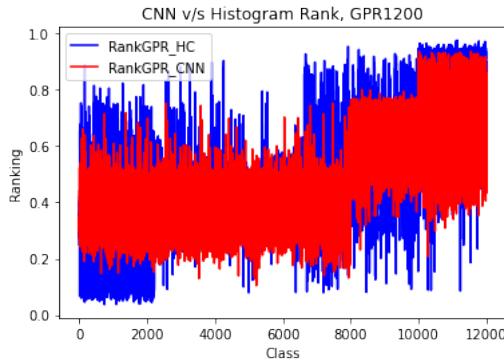


Figura 6: Gráfico comparativo entre el ranking obtenido para ambos métodos de extracción

En la sección 7 se comenta más sobre esto.

5. Algoritmo de búsqueda

A continuación, se detalla el funcionamiento del algoritmo de búsqueda implementado para las imágenes. Este, se puede ver implementado en los archivos mencionados en la sección 10, el cual corresponde a la unión entre las funciones rank normalizado y plot rank.

- En primer lugar, el usuario le entrega a la función plot rank el nombre de una imagen de consulta en formato de string, como por ejemplo:

```
1  plot_rank('122900') # Para INRIA
2  plot_rank('1009_nm0103797_rm3920397568_1983-4-15_2011') # Para GPR1200
```

Con esto, el algoritmo tardará entre 7 y 10 segundos en realizar los cálculos pertinentes y realizar un plot a la imagen consultada con su respectivo nombre, para luego realizarlo con las imágenes posicionadas según el ranking.

- El trasfondo de esta función, es que toma el nombre de la imagen de consulta y le aplica la función rank normalizado, devolviendo, el valor del Rank según la ecuación 2, una lista con los nombres de las 10 imágenes más similares y la cantidad de imágenes de la misma clase presentes en la base de datos.
- Con los valores extraídos del punto anterior, el algoritmo comienza a realizar el plot en orden siguiendo los nombres de la lista entregada por la función.

6. Resultados

6.1. INRIA Holidays database

A continuación, se muestran algunos ejemplos cualitativos del desempeño del algoritmo para la base de datos de INRIA.

Utilizando la siguiente imagen de consulta.



Figura 7: Imagen consultada para ambos extractores

Se obtuvieron los siguientes resultados para el método clásico de extracción.

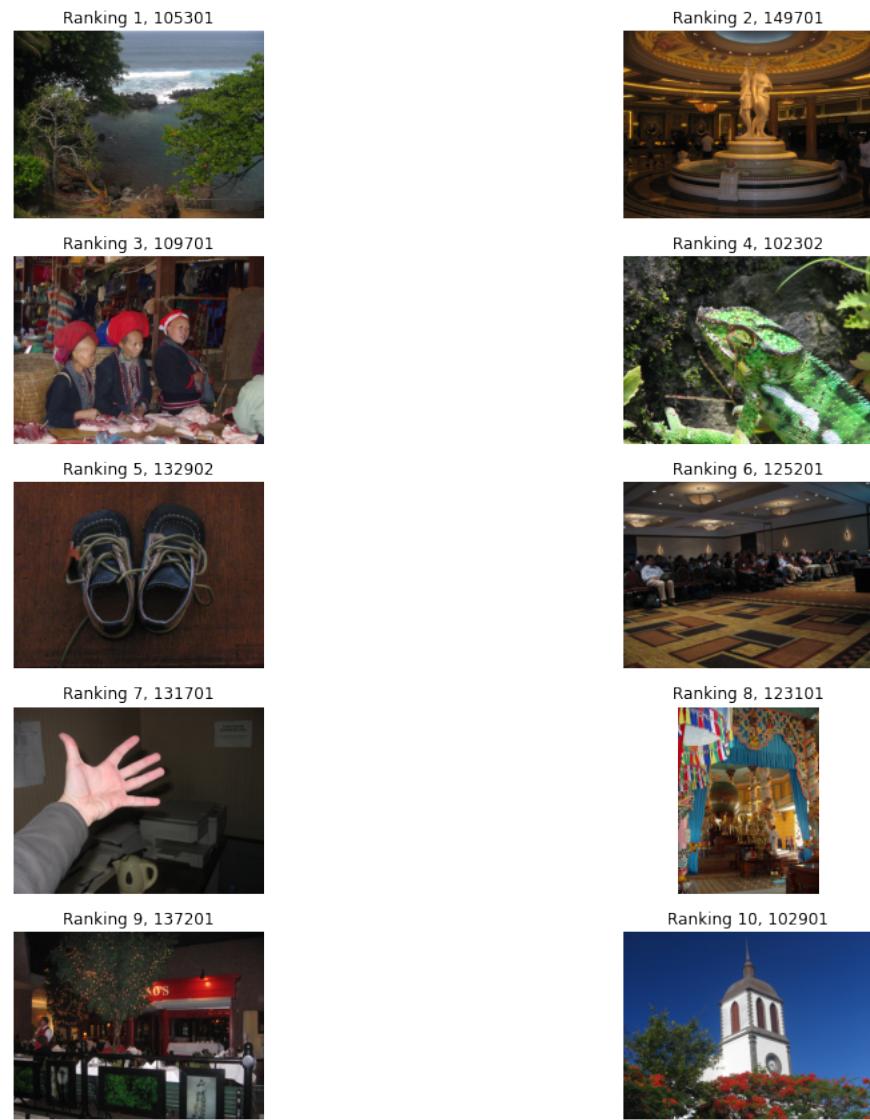


Figura 8: Resultados usando el método clásico

En la figura anterior, se puede observar que el algoritmo no entrega ninguna imagen de la misma clase, logrando 0 de 1 aciertos. Por otro lado, para el método CNN se obtuvo lo siguiente.

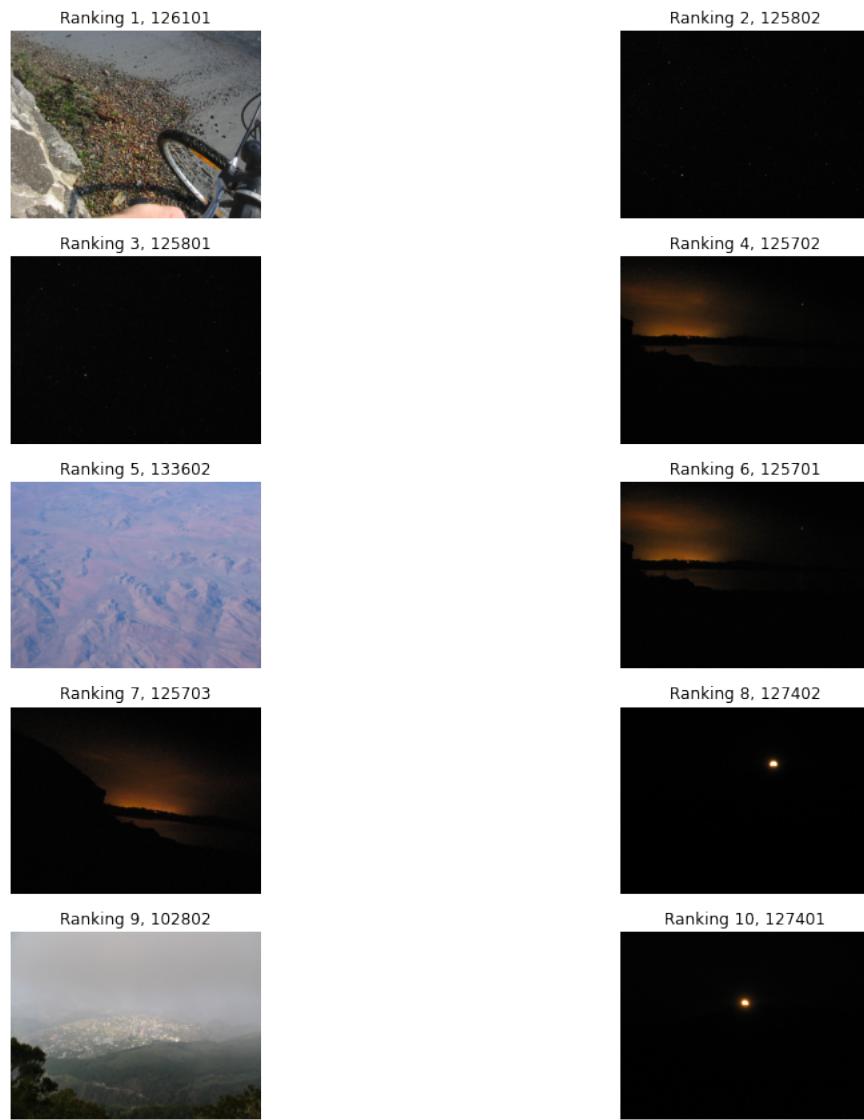


Figura 9: Resultados usando el método CNN

Observando que logra clasificar la única imagen de igual clase existente en la base de datos y en primera posición, logrando un excelente desempeño.
No obstante, al ocupar la siguiente imagen de consulta.

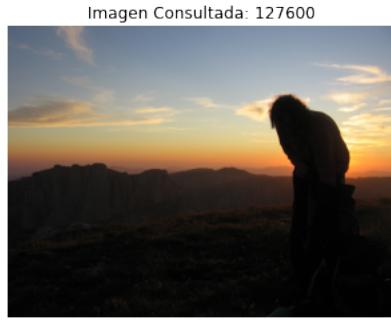


Figura 10: Imagen consultada para ambos extractores

Se obtienen los siguientes resultados para el método clásico.

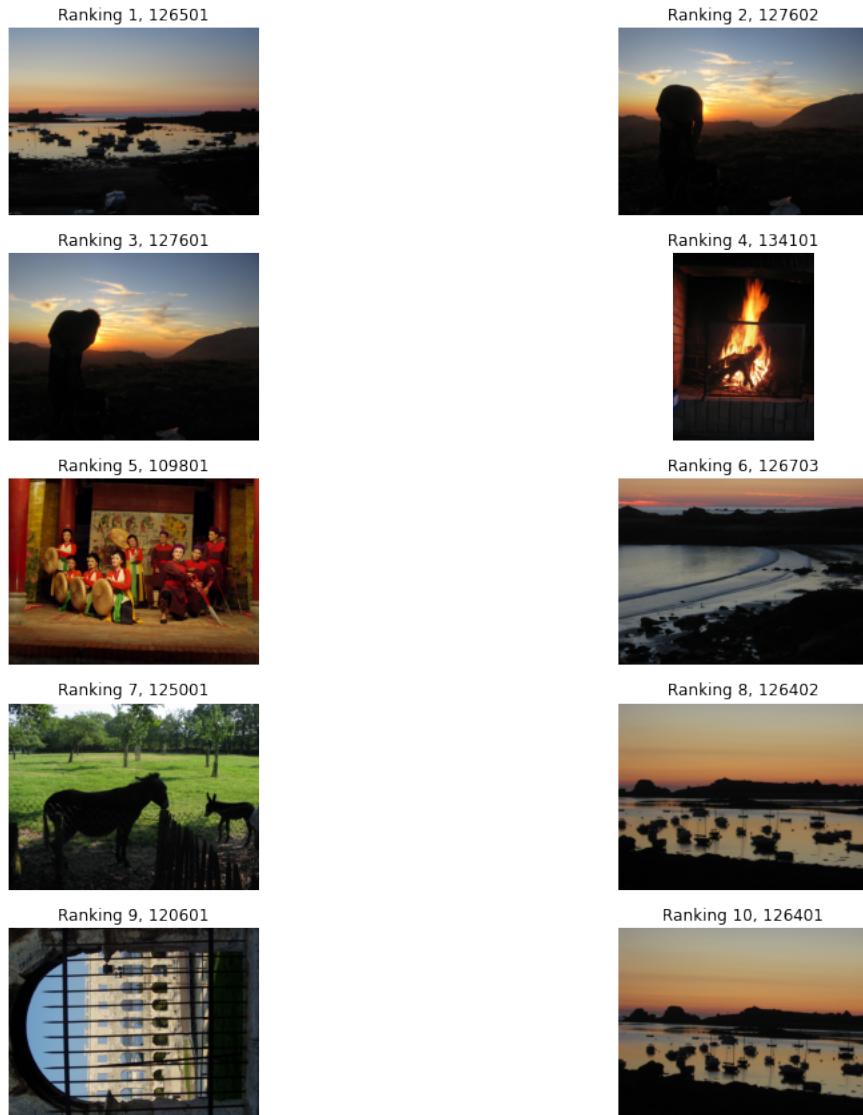


Figura 11: Resultados usando el método clásico

Notando que se clasifican correctamente las dos imágenes de igual clase existentes y en las posiciones 2 y 3 respectivamente, además, hay más imágenes que se asemejan en cuanto a colores o patrones en su forma. Por otro lado, para esta misma imagen, los resultados del extractor CNN se presentan a continuación.

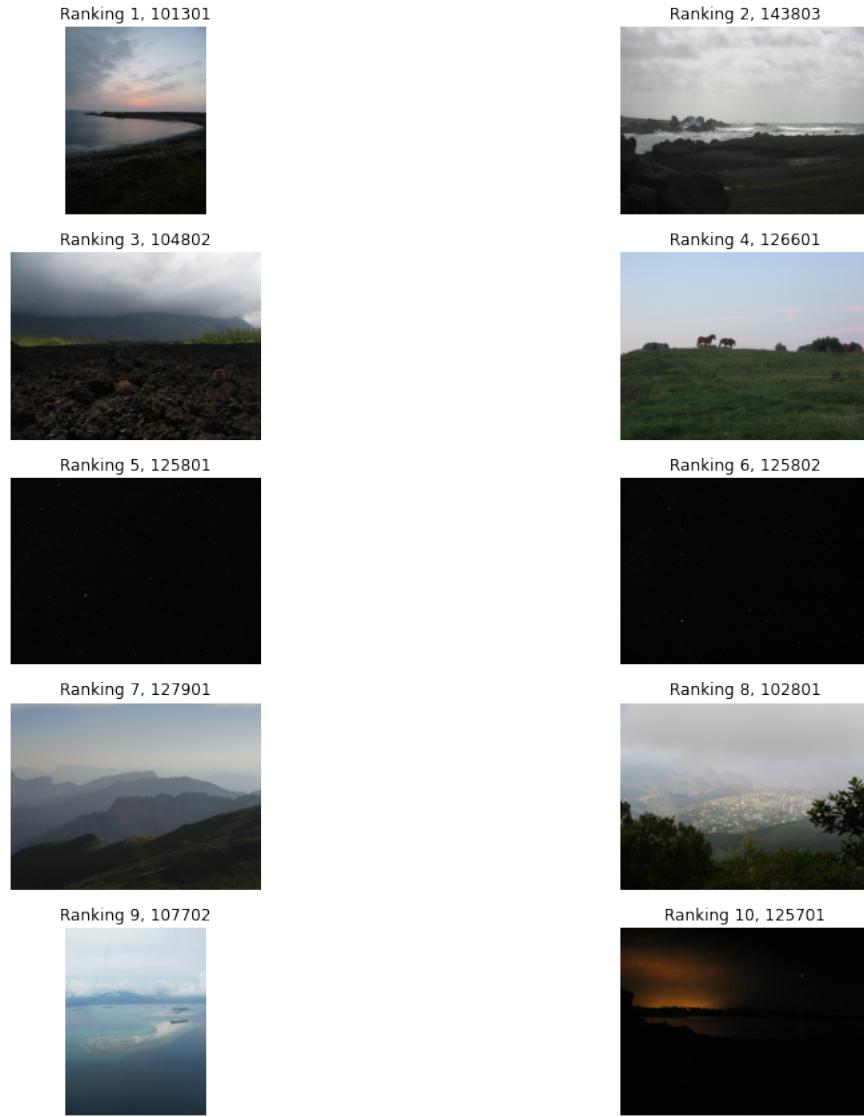


Figura 12: Resultados usando el método CNN

Donde se puede observar que no clasifica correctamente ninguna imagen, más aun, las imágenes entregadas se asemejan menos que en el caso anterior. Como se verá en la sección 7, estos ejemplos visuales no bastan para medir el desempeño del algoritmo y es por esto que se utilizará la medida del ranking normalizado.

6.2. GPR1200

A continuación, se muestran algunos ejemplos cualitativos del desempeño del algoritmo para la base de datos de GPR1200.

Usando la siguiente imagen de consulta.

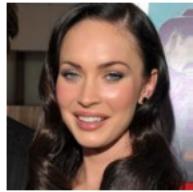
Imagen Consultada: 1198_nm0004874_rm344496128_1967-7-18_2004



Figura 13: Imagen de consulta para ambos extractores

Se obtuvieron los siguientes resultados para el extractor clásico.

Ranking 1, 1140_nm1083271_rm4039410432_1986-5-16_2010



Ranking 2, 1144_nm1218924_rm1014992640_1978-8-19_2009



Ranking 3, 1177_nm0906547_rm1556781056_1973-7-22_2005



Ranking 4, 1123_nm0513964_rm3520829440_1967-9-23_2002



Ranking 5, 1023_nm0957909_rm2651559424_1969-6-28_2007



Ranking 6, 1138_nm0001884_rm3085671168_1929-4-10_2001



Ranking 7, 1006_nm0330722_rm2150406912_1948-3-31_2007



Ranking 8, 1199_nm1579753_rm3575486976_1982-9-4_2004



Ranking 9, 1082_nm0000116_rm3840120832_1954-8-16_2006



Ranking 10, 1146_nm1335291_rm1991104512_1982-6-16_2010



Figura 14: Resultados obtenidos para el método clásico

Donde se puede observar que no acierta en ninguna imagen, más aun, entrega resultados femeninos. Por otro lado, para el extractor CNN.

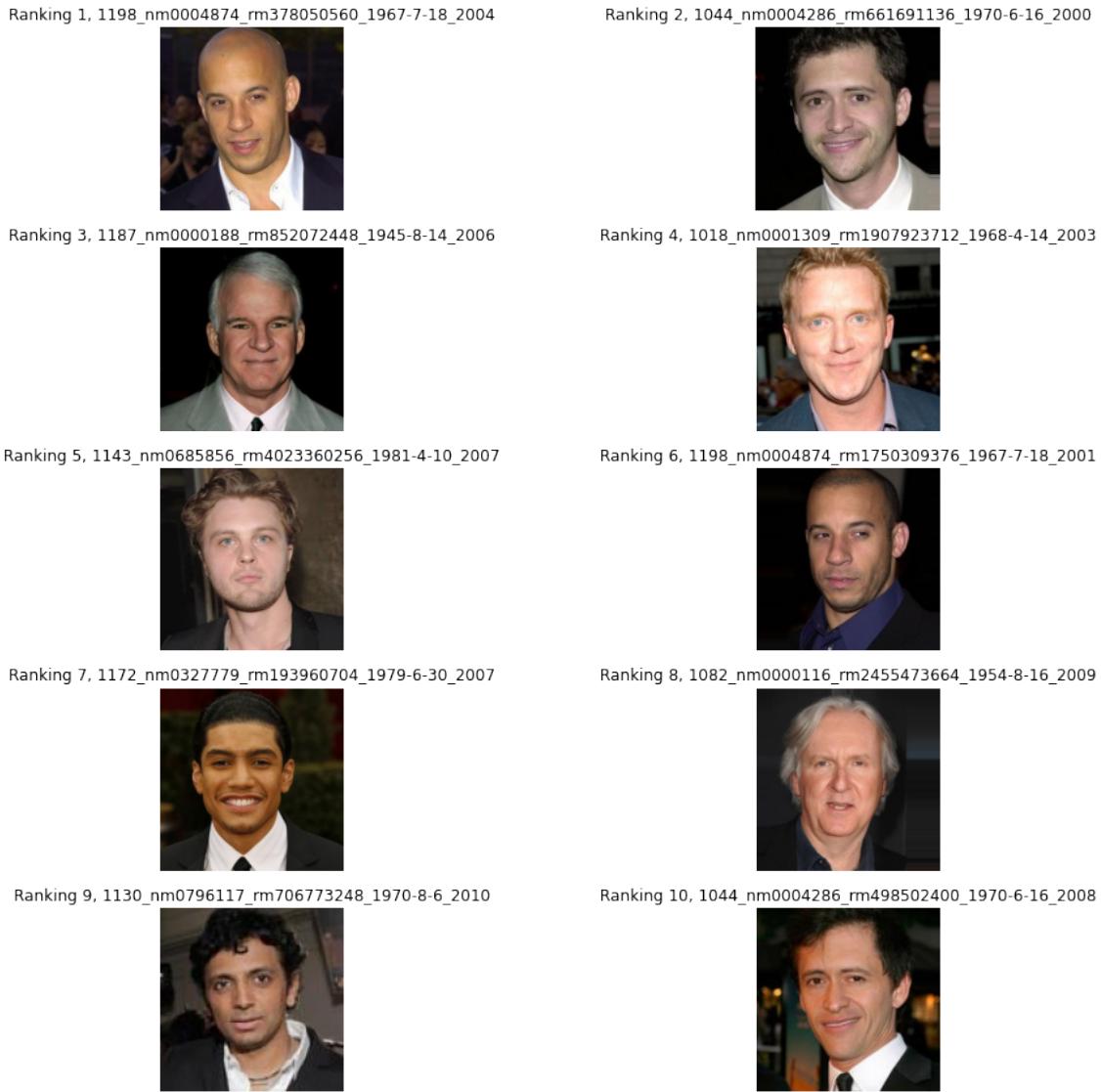


Figura 15: Resultados obtenidos para el extractor CNN

Donde se visualizan dos aciertos (1 y 6) y además, el algoritmo arroja resultados únicamente masculinos, por lo que funciona de mejor manera para esta imagen en particular.

Analizando una imagen distinta, esta vez correspondiente a un objeto con fondo blanco, como se muestra a continuación.

Imagen Consultada: 911_181796607370_6



Figura 16: Imagen de consulta para ambos extractores

La cual corresponde a un sillón en un fondo blanco, para esta, se obtuvieron los siguientes resultados con el extractor clásico.



Figura 17: Resultados obtenidos para el extractor clásico

¹ En la imagen anterior, se observa que usando el método clásico se obtienen 4 aciertos correctos, pero que los demás resultados no se acercan a lo consultado. Por otro lado, en los resultados con el extractor CNN que se muestran a continuación, se puede observar que el algoritmo detecta correctamente 4 sillones de la misma clase y los otros 6 resultados corresponden a un sillón parecido al consultado, evidenciando un mejor desempeño para esta imagen.

¹ A pesar de graficar con plt.axis('off') una imagen salió con ejes.

Ranking 1, 911_181796607370_6



Ranking 2, 911_181796607370_5



Ranking 3, 911_181796607370_2



Ranking 4, 911_181796607370_9



Ranking 5, 910_181795062950_3



Ranking 6, 910_181795062950_6



Ranking 7, 910_181795062950_9



Ranking 8, 910_181795062950_0



Ranking 9, 910_181795062950_11



Ranking 10, 910_181795062950_2



Figura 18: Resultados obtenidos para el extractor CNN

Si bien, con los ejemplos anteriores se puede tener una noción de que extractor funciona de mejor manera, esto se debe corroborar con los resultados de rank normalizado, no obstante, en los Notebooks entregados existen más ejemplos de la función plot_rank, con su respectivo número de aciertos.

7. Análisis de resultados

Para realizar el análisis de los resultados, se puede abordar el problema desde dos enfoques distintos.

7.1. Análisis Cualitativo

En este punto, entran en juego los ejemplos de consulta mostrados en la sección 6, donde se puede observar, que dependiendo de la imagen, puede ser mejor un método u el otro, es decir, que en algunas imágenes se podría pensar que al extraer características con un método clásico se obtienen mejores resultados o viceversa. No obstante, esta idea se debe corroborar usando el rank normalizado y lo analizado en la sección 4. No obstante, ambos algoritmos clasifican imágenes logrando un buen desempeño, acertando a la totalidad de clases o solamente en algunas de ellas.

7.2. Análisis Cuantitativo

En este caso, se deben analizar los resultados con respecto al ranking normalizado obtenido para ambas bases de datos. En primer lugar, observando la tabla 1 correspondiente a la base de datos INRIA, se puede notar que al extraer características usando la red pre-entrenada se obtienen mejores resultados, ya que su ranking normalizado es mejor 325 veces, en comparación a las 134 del método clásico.

Por otro lado, para la base GPR1200, se puede observar en la tabla 2 que se obtienen mejores resultados con el extractor de características CNN.

8. Optimización

Para optimizar los tiempos de búsqueda, se analizan diferentes opciones dentro de las cuales se encuentran:

Para optimizar el calculo de la distancia, la cual tiene un se demora bastante cuando se comparan las features de la red VGG16 (vectores de largo ≈ 25000), se propone utilizar la función de distancia euclíadiana que contiene la librería Scipy pues está optimizada y codificada de forma de realizar el calculo en un menor tiempo. A continuación se presenta la nueva función de distancia.

```

1 from scipy.spatial import distance
2
3 def distancia_euclidiana_optimizada(x,y):
4     """
5         Medida de similitud entre vectores de características,
6         usando la distancia euclíadiana.
7
8         :param numpy.ndarray x: primer vector
9         :param numpy.ndarray y: segundo vector
10
11        :return: la distancia euclíadiana entre los vectores
12        """
13
14        return distance.euclidean(x,y)

```

Se comparan los tiempos de computo de ambas funciones los cuales se muestran en la tabla 3. Como se puede observar en esta tabla, **el tiempo promedio que toma la función distancia optimizada es 836 veces menor al tiempo promedio de la distancia normal**, convirtiéndose en una gran optimización de tiempo al comparar las features de la imagen con la base de datos y reduciendo considerablemente el tiempo de ejecución del algoritmo.

Además, como es una optimización con respecto al tiempo de computo de las distancias y no de la

Tabla 3: Tiempo promedio y desviación en determinar la distancia euclíadiana entre 2 vectores de largo 25088 para el caso normal y el caso optimizado (tomando 100 muestras de tiempo).

| | Tiempo de computo [segundos] |
|----------------------|------------------------------|
| Distancia normal | 0.0836 ± 0.0046 |
| Distancia optimizada | 0.0001 ± 0.0002 |

dimensión de los vectores, el algoritmo sigue funcionando de igual manera, por lo que los resultados no se ven afectados por este cambio.

9. Conclusión

Se logró desarrollar e implementar un algoritmo de CBIR para ambos conjuntos de datos, cumpliendo así el objetivo principal del proyecto. Esto permitió aterrizar distintos conceptos vistos tanto en cátedra como en cursos anteriores, esto otorga una gran experiencia práctica al equipo de trabajo, lo cual era uno de los objetivos generales del curso. Además, fue posible comparar el método clásico con una red CNN, logrando distinguir sus ventajas y desventajas en el proceso de extracción de características.

Dentro de las dificultades que hubieron dentro del desarrollo del proyecto se encuentran los largos tiempos de ejecución de los códigos, los cuales se desconectaban en algunas ocasiones cuando se trabajaba en la plataforma de Google Colab, problema que retrasó bastante el avance de estos, sin embargo, lo anterior fue solucionado al trabajar de forma local con Jupyter Notebook. Otra dificultad que fue considerable al momento de desarrollar el proyecto, fue una mala comprensión del Rank Normalizado, lo cual se logró corregir a tiempo y ajustar debidamente las funciones para que el algoritmo funcionara como lo esperado.

También, es importante destacar otras posibles mejoras que podrían realizarse a futuro para mejorar el funcionamiento del algoritmo. Por ejemplo, en vez de intentar guardar todo un conjunto de vectores de distancias, cada vez que se consulte por una imagen, dejar registro de esto y en caso de realizar la misma consulta en algún momento, acceder más fácil a esta, requiriendo menos recursos. Junto a esto, se destaca lo importante que es este tipo de algoritmos en la actualidad, ya que las personas y su entorno ha emigrado cada vez más al “mundo digital”, por lo tanto, aprender sobre este tipo de procesamiento de imágenes es una herramienta fundamental que no solamente se puede aplicar a la búsqueda de estas, si no que también a problemáticas más complejas que requieran de un análisis exhaustivo de las características extraídas.

Referencias

- [1] R. Thakur. Step by step VGG16 implementation in Keras for beginners. Disponible en: <https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c>
- [2] F. Chollet. Transfer learning & fine-tuning. Disponible en: https://keras.io/guides/transfer_learning/
- [3] Image processing in Python. Disponible en: <https://scikit-image.org/>

10. Anexos

10.1. Repositorio del Curso

En el siguiente enlace https://github.com/BenjaminIrarrazabal/Laboratorios_Inteligencia, se puede encontrar el repositorio de GitHub trabajado durante el semestre, propiedad de ambos integrantes del grupo, Benjamín Irarrázabal y Joaquín Zepeda. En este se encuentran los Notebooks de las experiencias de todo el semestre, junto a otros archivos importantes.

10.2. Instrucciones Proyecto

En esta sección se explicará más detalladamente sobre las instrucciones del proyecto, el proceso para la extracción de características y el posterior procesamiento realizado para implementar el algoritmo y las otras medidas encontradas. Cabe destacar que si bien todos los códigos son entregados en conjunto por la plataforma U-Cursos, se puede acceder a ellos desde el link de GitHub dejado en la sección anterior.

En el siguiente punteo, se detallan las instrucciones correspondientes.

- En primer lugar, se debe tomar el Notebook CBIR feature extractor con el cual se puede observar el código utilizado para extraer características. Este es necesario correrlo sólo una vez, ya que las características están guardadas en el Repositorio GitHub (Revisar Laboratorios_Inteligencia/Proyecto Final/features para ver los archivos pkl). Esto con excepción de las features de GPR1200 extraídas con el método CNN ya que el archivo pesa más de 1GB y sólo fue trabajado de manera local con Jupyter Notebook.
- Luego, con los vectores de características ya creados, se puede continuar con los archivos CBIR GPR1200 HC, CBIR GPR1200 CNN, CBIR INRIA Holidays database HC, CBIR INRIA Holidays database CNN.
- Es importante destacar que los códigos son casi idénticos para ambas bases de datos, solamente cambian algunos detalles con respecto al recorrido de las bases de datos y de cómo se analizan las clases debido al formato de los nombres.
- Junto a esto, se puede tomar el Notebook Rank Comparison, que toma los frames de Ranking Normalizado entre ambos métodos para las dos bases de datos y realiza las comparaciones de la sección 4.
- Finalmente, para correr cada Notebook solo debe seguir paso por paso cada celda. En las últimas se encontrarán los ejemplos cualitativos de los cuales se extraen imágenes para el informe.