Laboratorio de Análisis y Procesamiento de Imágenes
Semana 12 y 13
Fecha de realización: 17 de abril - 2 de mayo de 2020

Universidad de los Andes

Práctica 7:

K-Means y Textones

1. Objetivos

- Clasificar imágenes utilizando *kmeans*.
- Estudiar las propiedades de *kmeans*.
- Estudiar el descriptor de textura textones.
- Aprender a segmentar por Watersheds con y sin marcadores.

2. Parámetros de entrega

Sicua

- Adjuntar un único archivo (a menos que se especifique lo contrario) con los códigos de la siguiente forma: *main_CódigoEstudiante#1_CódigoEstudiante#2.m* en Matlab o
  main_CódigoEstudiante#1_CódigoEstudiante#2.py en Python.
- Adjuntar **TODO** el código que se entregó en el item anterior, en formato *.txt*. Llámelo de igual manera: *CódigoEstudiante#1_CódigoEstudiante#2.txt*. Esto con el fin de poder evaluar en Sicua automáticamente cualquier intento de copia o similitud entre los algoritmos. Cualquier intento de copia será tratado de acuerdo al reglamento de la Universidad. **Aquel grupo que no incluya este item en su entrega tendrá una nota de 0 en el laboratorio**.
- Adjuntar un único archivo con el informe en PDF con el mismo nombre del primer item: *CódigoEstudiante#1_CódigoEstudiante#2.pdf*. El informe debe presentarse con el formato CVPR (para más información del informe ver sección de Parámetros de calificación).
- En el caso de que no tenga compañero, **DEBE** utilizar un cero como el código de su compañero, i.e. *main_CódigoEstudiante_0.m* para el archivo de matlab, *CódigoEstudiante_0.pdf* para el informe y *CódigoEstudiante_0.txt* para el archivo de texto.
- **NO** se permiten archivos comprimidos tales como *zip, rar, 7z, tar, gz, xz, iso, cbz*, etc. Aquel grupo que envíe su informe como un archivo comprimido no tendrá calificación.

- No se recibirá ningún archivo por algún medio diferente a Sicua plus.
- Tendrán una semana para realizar cada práctica (Informe, código, etc.). La guía se entregará el viernes anterior a la sesión y el plazo de entrega será hasta las 11:59 pm del sábado de la semana de la sesión. El vínculo para el envío del laboratorio dejará de estar disponible luego de esta hora. **No se recibirá ningún laboratorio después de la hora de entrega** (la hora queda registrada en el envío de Sicua). Cada estudiante tiene 2 intentos en Sicua para cada entrega, pueden hacer entregas parciales y se calificará el último intento.

3. Algunas reglas

- La asistencia a la sección de laboratorio inscrita es **obligatoria**. Acorde con el Reglamento General de Estudiantes de Pregrado de la Universidad de los Andes, la inasistencia a más del 20 % de las clases de laboratorio resultará en la reprobación de la materia completa (laboratorio y magistral).
- Los informes deben realizarse **únicamente** con la pareja. Esto quiere decir que aunque es válido discutir los problemas con sus compañeros, la solución y el código, deben ser de su completa autoría. Está prohibido copiar literalmente el algoritmo y/o procedimiento desarrollado por otro grupo. Si llega a obtener un código de internet, asociado al problema a resolver, este debe estar debidamente referenciado y usted debe entenderlo por completo.

4. Parámetros de calificación

Resultados

1. Todos los códigos deben mantener orden y coherencia en la ejecución de comandos, es decir, cada vez que se muestre una figura, el programa debe esperar para que se presione una tecla, para así continuar con la siguiente y así sucesivamente (para esto utilice en Matlab `pause`) o en Python `input("Press Enter to continue...")`. Tenga en cuenta que si se quieren contrastar dos imágenes use `subplot` o `imshowpair`.
2. Toda figura debe llevar su título y descripción en el informe.
3. El código debe estar debidamente comentado.
4. NUNCA utilice rutas absolutas para leer o guardar archivos. Este es el error más común en la ejecución de los códigos.
5. Para generar rutas utilice `fullfile` en Matlab o `os.path.join` en Python ya que puede que corramos los laboratorios usando Linux o Windows y los separadores de archivos cambian dependiendo del sistema operativo.
6. Asuma que dentro de la carpeta de ejecución del código se encuentran los archivos necesarios para el laboratorio.

   Ejemplo: dentro del código principal el estudiante quiere leer la imagen `im.png` que está dentro de una carpeta de imágenes en la misma ruta que el main.

6.1. Forma incorrecta:
En Matlab:
`imread('C:/Estudiante/docs/ElLab/ims/im.png').`
En Python:
`skimage.io.imread('C:/Estudiante/docs/ElLab/ims/im.png')`

6.2. Forma correcta:
En Matlab:
`imread(fullfile('ims','im.png')).`
En Python:
`skimage.io.imread(os.path.join('ims','im.png'))`

Informe

Todos los laboratorios deben realizarse en formato CVPR. Si desean realizarlo en LaTeXo word, pueden obtener una plantilla del formato en el siguiente link. Cabe resaltar que los informes del laboratorio no deben contener ninguna sección de artículo científico, esto significa que no deben incluir ninguna división como resultados, abstract o conclusiones. Por consiguiente, **deben responder únicamente a las preguntas del informe.** Recuerden incluir imágenes de sus resultados y documentarlas debidamente. Por último, el informe tiene una longitud máxima de 3 páginas. Se pueden incluir imágenes como anexos pero las imágenes principales deben ser parte del informe.

Bonos

Cada grupo ganará puntos que le suben la nota por cada una de las siguientes características que se cumpla:

1. Usar Python, sea ya en *Notebooks* de *Jupyter*, o en scripts convencionales de Python.

2. Desarrollar el informe en LaTeX. (Aquellas personas que lo desarrollen en LaTeX, deben escribir al final del informe **"Realizado en LaTeX"**. De lo contrario no se contará el bono. Los grupos que intenten reproducir la frase en un informe realizado en Word tendrán 0 en la nota de dicho laboratorio. Para poder escribir el logo utilice el comando "\LaTeX" en su informe de latex.

Estos puntos se asignarán de acuerdo al criterio de las profesoras.

5. Informe

Para este laboratorio utilizaremos la base de datos que está en este link. El punto de Clasificación y el de Textones utilizarán las carpetas de train y val, mientras que el de Watersheds la carpeta de watersheds.

5.1. Clasificación de imágenes

Clasificaremos las imágenes haciendo uso del método clasificación no supervisada K-means. Para que K-means agrupe las imágenes en clusters, primero tenemos que representar las imágenes de alguna forma, es decir, cada imagen debe ser un punto en un espacio de dimensión superior. La representación para las imágenes será un histograma que concatene el histograma de los canales R, G y B en ese orden, es decir, si asumimos que tenemos 10 bins por canal, nuestra representación para una imagen es de tamaño 30 (10*3). Si está utilizando Python recomendamos la función de *k-means* de scikit-learn sklearn.cluster.KMeans.

1. Cree una funcione que reciba una imagen de entrada y produzca el histograma de color concatenado según un parámetro de entrada que indique el número de bins. Esta función debe normalizan el histograma según el número de pixeles en la imagen.

2. Visualice en un subplot el histograma de una imagen de entrenamiento de cada clase utilizando 2 valores diferentes de bins. Debería tener en total número de clases x 2 histogramas en su(s) figura(s). Asegúrese de ponerle un título a cada histograma indicando el número de bins y la clase de la imagen.

3. Según el subplot anterior elija un número de bins para su representación. En el informe especifique y justifique su elección (haga uso del subplot y la teoría de *k-means*).

4. Aplique *k-means* con 3 clusters sobre los histogramas de las imágenes con el número de bins que eligió anteriormente, de manera que encuentre la etiqueta que le corresponde a cada imagen según *k-means*. *Nota:* en la linea de código inmediatamente antes de aplicar *kmeans*, asigne un valor arbitrario a la semilla del generador de números pseudo-aleatorios, para reproducibilidad. En Python es `np.random.seed(semilla)`, y en Matlab es `rng(semilla)`. ¿Por qué se corre *k-means* con 3 clusters?

5. Visualice en un subplot las imágenes de entrenamiento junto con el cluster asignado por *k-means* y su histograma respectivo.

6. Reporte en el informe una tabla con la clase y el cluster que usted eligió para esa clase. *K-means* separa las imágenes en clusters, pero no necesariamente el cluster 1 corresponde a la clase 1, para asignar esa correspondencia utilice el subplot del numeral anterior, por ejemplo si la mayoría de las imágenes de la clase 1 están en el cluster 2, puede determinar que todas las imágenes que k-means indique pertenezcan al cluster 2 se les está asginarndo la etiqueta de la clase 1.

7. Utilice el modelo de *k-means* ya entrenado (no vaya a volver a entrenar) para predecir la clase de las imágenes de validación.

8. Calcule e imprima el ACA y la matriz de confusión de la clasificación en el conjunto de validación. Recuerde que para esto puede hacer uso de las funciones nativas de Python y Matlab.

9. ¿Cuál es la distancia que usa *kmeans*? y, por tanto, ¿qué forma geométrica es aquella que implícitamente busca en los clusters?

10. Dada la respuesta del punto anterior, comente en un párrafo del informe si cree que esta distancia tiene sentido al momento de comparar histogramas.

11. Reporte en un párrafo del informe algunas distancias que existen para comparar histogramas, es decir, distancias *entre* histogramas.

12. ¿Cómo fue el desempeño del método? ¿en qué falla? ¿es problema de los descriptores que escogimos? ¿es problema del algoritmo que escogimos (*kmeans*)? ¿ambos?

5.2. Textones

Para clasificar las imágenes por medio de histogramas de textones usted deberá crear las funciones enumeradas a continuación. Descargue el banco de filtros de este link.

1. Escriba una función llamada `CalculateFilterResponse_código1_código2(grayscale_image, filters)` que toma de entrada:

   - `grayscale_image` Una imagen en escala de grises.
   - `filters` Un banco de filtros para calcular la cross-correlación.

   La función retorna:

   - `resp` Una matriz con la respuesta de la cross-correlación de la imagen con todos los filtros. Esta matriz es de dimensión $M \times N \times$ número de filtros, donde M y N son las dimensiones de la imagen de entrada.

2. Escriba una función que calcule el diccionario de textones a partir de las respuestas de las imágenes de entrenamiento a la cross-correlación con el banco de filtros. Para realizar esta función haga uso de la función 1. Esta función se debe llamar: `CalculateTextonDictionary_código1_código2(image_names, filters, k, filename)` que toma de entrada:

   - `image_names` Una lista de las rutas de de las imágenes, la ruta debe incluir el nombre de cada imagen (carpetas + nombre).
   - `filters` Un banco de filtros para calcular la cross-correlación.
   - `k` El tamaño del diccionario de textones (el número de clusters para k-means).
   - `filename` el nombre del archivo .mat dónde va guardar el diccionario de textones (el modelo de k-means entrenado con las imágenes de entrenamiento). Este nombre debe ser `textonDictionary_código1_código2.mat`.

3. Haga uso del diccionario de textones obtenido en la función 2 para obtener el histogramas de textones de una imagen, para esto escriba una función llamada `CalculateTextonHistogram_código1_código2(grayscale_image, centroids, dist)` que toma de entrada:

   - `grayscale_image` Una imagen en escala de grises.
   - `centroids` Los centroides del diccionario de textones, se calcularon por medio de la función anterior.
   - `dist` La distancia para calcular el mapa de textones, esta distancia será utilizada para asignar a cada pixel en la imagen de entrada el textón más cercano.

   La función retorna:

   - `hist` El histograma de textones de la imagen de entrada.

4. Cree una función que asigne a una imagen de prueba su clase correspondiente por medio del método de vecino más cercano (NN). Para esto, haga uso de la función 3 para calcular el histograma de textones de su nueva imagen. Compare este histograma de textones con los histogramas de textones de las imágenes de entrenamiento. La función deberá llamarse `PredictClass_código1_código2(grayscale_image, filters, train_hists, centroids, dist, nn_dist)` y tiene como parámetros de entrada:

   - `grayscale_image` Una imagen en escala de grises.
   - `filters` Un banco de filtros para calcular la cross-correlación.
   - `train_hists` Una matriz con los histogramas de textones de las imágenes de entrenamiento. Esta matriz es de dimensión número de imágenes x bins del histograma de textones.
   - `centroids` Los centroides del diccionario de textones, se calcularon por medio de la función anterior.
   - `dist` La distancia para calcular el mapa de textones, esta distancia será utilizada para asignar a cada pixel en la imagen de entrada el textón más cercano.
   - `nn_dist` La distancia para asignar el vecino más cercano (es recomendable definir una distancia de histogramas).

   La función retorna:

   - `class` La clase a la que pertence la imagen de entrada.

5.2.1. Preguntas

Deben responder estas preguntas en el informe escrito:

- Reporte el ACA de su método obtenido en el dataset de prueba y muestre el subplot generado con las imágenes y sus respectivas etiquetas.

- ¿Si se deseara realizar un banco de filtros desde cero, qué procedimiento se debería seguir? Explique este procedimiento por medio de un diagrama de flujo.
- Explique qué es un texton en máximo un párrafo.
- Enumere tres posibles aplicaciones biomédicas del uso de textones.

5.3. BONO: Watersheds (+0.5)

El bono en la nota se otorgará únicamente a trabajos que tengan este punto completo. En este punto el objetivo es segmentar glándulas en una imagen histológicas. La imagen y su anotación se encuentran en la carpeta *watersheds* de la carpeta de datos utilizada en este laboratorio. Si están programando en Python, les recomendamos utilizar las funciones de morfología del paquete *skimage.morphology* y la de watersheds de *skimage.segmentation*. Recuerde nombrar todos las imágenes que muestra en los subplots.

1. Obtenga el gradiente morfológico de la imagen en escala de gris (no debe ser confundido con el gradiente obtenido por medio de cross-correlación). Es recomendable utilizar un elemento estructurante tan grande como para cubrir el centro del cúmulo de glándulas. Reporte en el informe los parámetros y funciones utilizadas para este numeral.
2. Utilice watersheds sin marcadores sobre la imagen del gradiente morfológico para segmentar las glándulas.
3. Visualice en un subplot la imagen original, el gradiente morfológico, el resultado de hacer watersheds sin marcadores y la anotación. ¿Cuál es el efecto de emplear watersheds sin marcadores?
4. Binarice la imagen y utilice operaciones morfológicas para obtener en su mayoría las glándulas y eliminar los otros pixeles del fondo. Reporte en el informe los parámetros y funciones que utilizó para este proceso.
5. Utilice una función de *h_minima* como la de *skimage.morphology* para conseguir los mínimos de la imagen mayores a un umbral que usted determine. Dilate y etiquete los componentes conexos que obtuvo como resultado de la operación anterior. Los componentes conexos etiquetados son los marcadores para la imagen. Reporte en el informe el umbral elegido y los parámetros de la dilatación.
6. Aplique watersheds al gradiente morfológico utilizando los marcadores y como máscara la imagen binaria.
7. Visualice en un subplot la imagen en escala de gris, la imagen binaria, el gradiente morfológico, los marcadores y la segmentación haciendo uso de watersheds con marcadores.
8. Calcule y reporte el índice de Jaccard de su mejor segmentación.

5.4. Entregables

En este laboratorio deberá entregar:

1. El informe del laboratorio `Código1_Código2.pdf`.

2. Un archivo `mainKmeans_Código1_Código2.m` o `.py` que contenga la función para generar el histograma de color concatenado además del código para aplicar *k-means* especificado en el punto 5.1. Este código también debe visualizar los subplots requeridos, imprimir el ACA y la matriz de confusión para su problema.

3. Un archivo llamado `textonDictionary_Código1_Código2.mat` que corresponde al diccionario de textones generado con $k = 3$ textones.

4. Un archivo `mainTextons_Código1_Código2.m` o `.py` que contenga el código para entrenar y clasificar las imágenes de entrenamiento y prueba por medio del método de histograma de textones. Este código también debe visualizar en un subplot las imágenes con sus etiquetas e imprimir en pantalla el ACA del método.

   Tenga en cuenta que este archivo **NO DEBE GENERAR EL DICCIONARIO DE TEXTONES**, debe tener comentado el código para generar el diccionario pero NO debe ejecutarlo. Este archivo debe cargar el diccionario de textones con $k = 3$ textones que adjunta como entregable y a partir de este, clasificar las imágenes de la base de datos.

5. Un archivo .txt que deberá contener el código para **TODO** el laboratorio en **UN SOLO ARCHIVO**.

6. (opcional) Un archivo `mainWatersheds_Código1_Código2.m` o `.py` que contenga el código correspondiente al bono. El código debe mostrar los subplots requeridos en el enunciado.