TRABAJO FINAL – PARTE 2

1.1. Corte Normalizado

¿Qué hace este algoritmo?

Este algoritmo muestra cómo transformar una imagen en una versión segmentada de sí misma y cómo estas técnicas pueden mejorar la percepción de las regiones dentro de la imagen.

Este algoritmo utiliza dos enfoques diferentes basados en la agrupación de píxeles similares y luego compara los resultados. El ejemplo concreto utiliza una imagen de café proporcionada por la biblioteca scikit-image.

¿Cómo lo hace?

1. Carga de la Imagen: La imagen de café se carga utilizando la función data.coffee() de scikit-image.

2. Segmentación SLIC: La primera técnica de segmentación utilizada es SLIC (Simple Linear Iterative Clustering), que es una variante del algoritmo k-means específicamente adaptada para segmentar imágenes. La función segmentation.slic() recibe la imagen, el grado de compactación (compactness) que influye en cómo de espacialmente compactos serán los superpíxeles, y el número de segmentos o superpíxeles (n_segments) deseado. Esta función devuelve una matriz donde cada píxel tiene asignado el índice de su superpíxel correspondiente. Luego, color.label2rgb() se utiliza para visualizar el resultado de la segmentación, promediando los colores dentro de cada superpíxel.

3. Construcción del Grafo RAG: Posteriormente, se construye un grafo de adyacencia regional (RAG, por sus siglas en inglés) utilizando graph.rag_mean_color(), que conecta los superpíxeles de la imagen original basándose en la similitud de sus colores medios. El grafo resultante representa la estructura de la imagen de manera que los nodos corresponden a los superpíxeles y las aristas reflejan la similitud entre los colores de estos.

4. Corte Normalizado del Grafo: El paso siguiente implica aplicar el corte normalizado del grafo mediante graph.cut_normalized(), un método para particionar el grafo en subgrafos minimizando un valor (la suma normalizada de las diferencias entre grupos de nodos), lo que resulta en una nueva segmentación que busca ser más coherente en términos de color y textura. Esta técnica es especialmente útil para mejorar la cohesión de los segmentos en la imagen.

5. Visualización: Finalmente, el código utiliza matplotlib para mostrar la imagen original segmentada por SLIC y la imagen después de aplicar el corte normalizado del grafo. Se configuran los ejes para no mostrarlos y se ajusta el layout para una mejor presentación.

Desde un punto de vista matemático, este proceso involucra varios conceptos clave:

- Clustering k-means en SLIC: Optimización iterativa para minimizar la suma de distancias cuadradas entre los colores de los píxeles y los centros de sus respectivos superpíxeles.
- Grafos y corte normalizado: Modelado de la imagen como un grafo y búsqueda de una partición que minimice la suma de diferencias entre los grupos de nodos, ajustada por el tamaño de los grupos, para preservar la coherencia en la segmentación.

1.2. El algoritmo GrabCut

¿Qué hace este algoritmo?

Es un método de extracción de primer plano (foreground) en imágenes, diseñado para separar de forma eficaz objetos de interés del fondo con mínima intervención del usuario. Este proceso se basa en la modelización de las características de color de los píxeles mediante Modelos de Mezcla Gaussiana (GMM por sus siglas en inglés) y la minimización de una función de energía que define cuán bien se segmentan los píxeles en primer plano y fondo.

¿Cómo lo hace?

1. Entrada del Usuario: El usuario inicialmente proporciona una aproximación burda de la región de interés (ROI) dibujando un rectángulo alrededor del objeto que desea extraer. Opcionalmente, el usuario puede refinar la segmentación proporcionando marcas adicionales que indican áreas incorrectamente clasificadas como fondo (con marcas blancas) o como primer plano (con marcas negras).

2. Modelización Inicial: Dentro del rectángulo, los píxeles se consideran inicialmente como una mezcla de primer plano probable y fondo probable. Los píxeles fuera del rectángulo se consideran fondo seguro. Esta etapa inicial utiliza Modelos de Mezcla Gaussiana para modelar las distribuciones de color del primer plano y el fondo.

3. Asignación de Etiquetas: Se asigna una etiqueta a cada píxel basándose en la comparación de sus características de color con los modelos de GMM del primer plano y fondo. Los píxeles pueden ser etiquetados como primer plano seguro, fondo seguro, primer plano probable o fondo probable.

4. Construcción del Grafo y Minimización de la Energía: Se construye un grafo donde cada píxel es un nodo conectado a dos nodos especiales: una fuente (que representa el primer plano) y un sumidero (que representa el fondo). Los pesos de las aristas entre los píxeles y estos nodos especiales se basan en las probabilidades de que el píxel pertenezca al primer plano o al fondo, calculadas con los modelos GMM. Las aristas entre píxeles vecinos tienen pesos basados en la similitud de color y borde, promoviendo la coherencia en la asignación de etiquetas entre píxeles cercanos.

5. Corte de Grafo (Graph Cut): Se aplica un algoritmo de corte de grafo (por ejemplo, el algoritmo de Ford-Fulkerson) para encontrar la partición del grafo que minimiza la función de energía total. Esta partición separa los píxeles en primer plano y fondo con un costo mínimo, basado en la suma de los pesos de las aristas cortadas.

6. Iteración: El proceso puede iterarse, refinando los modelos GMM y la segmentación, hasta alcanzar una convergencia donde la clasificación de píxeles ya no cambia significativamente.

Desde un punto de vista matemático, la minimización de la función de energía, a través del corte de grafo, produce la segmentación final del primer plano y fondo, donde se ha encontrado el equilibrio entre ajustarse a los modelos de color y mantener la coherencia espacial en la imagen.