Informe BSDS: Visión por Computador

Emilio Botero
Universidad de los Andes
e.botero10@uniandes.edu.co

1.. Métodos de segmentación

Uno de los problemas principales en la visión artificial es obtener representaciones de imágenes que resumen o agrupan contenido en propiedades comunes. A esto se le conoce como segmentación. El clustering es una categoría de métodos que permiten lograr este objetivo al agrupar una imagen en clusters que se distinguen unos de otros por una serie de características o descriptores definidos según la aplicación. Dos métodos populares de clustering son kmeans y Gaussian Mixture Models. En ambos se obtienen descriptores de la imagen, y a partir de ellos se aplica un algoritmo que agrupa los descriptores en un set de clusters predefinido [1]. Estos descriptores contienen todas las medidas que se consideran relevantes para describir un píxel o una ventana de una imagen. Cada descriptor en una imagen, entonces, pertenece a un cluster, de tal manera que cada cluster representa un segmento de la imagen.

En Kmeans se parte de unos centroides inicializados de manera aleatoria en el espacio de representación. Los datos se asignan a los centroides según el centroide más cercano, donde la distancia utilizada puede ser euclidea o no. Una vez se tienen todos los datos asignados a un cluster, se recalculan los centroides de los clusters: la media de los datos en ese cluster es el nuevo centroide, y se repite la asignación de datos a centroides y cálculo de centroides hasta converger. Kmeans tiene varios problemas: tiende a agrupar datos en clusters esféricos y puede converger a mínimos locales. Los Gaussian Mixture Models pretenden solucionar esto al permitir que los centroides tengan formas no esféricas, concibiendo los clusters como componentes de una Mezcla de Gausianas multivariadas, de tal manera que se pueden elongar en sus ejes según la matriz de covarianza de cada componente.

Así, los Gaussian Mixture Models (GMM) parten de una asignación de los datos a componentes individuales. Los parámetros de cada componente se vuelven a estimar en un paso conocido como E-step (por expectation), y posteriormente se reasignan los datos a cada componente en un M-step (por maximization), y se repite lo anterior hasta converger. Este algoritmo también puede converger a mínimos locales.

Por su popularidad y su facilidad de implementación, se escogieron estos dos métodos para aplicarse a la base de datos BSDS [2], [3].

Teniendo en cuenta lo anterior, la pregunta es entonces ¿cómo escoger los descriptores? Una posible alternativa es hacer clustering según la distribución de color en la imagen. Las imágenes a color se representan en espacios de color compuestos típicamente por 3 dimensiones: una imágen representada por el modelo RGB consiste en tres planos 2-D, cada uno correspondiendo a un color primario. Estos planos son monocromáticos, es decir, representan las intensidades de los niveles de gris en esos componentes. Un monitor RGB puede entonces combinar estos planos para producir la imagen RGB. No obstante lo anterior, el modelo de color RGB no es muy adecuado para la percepción humana. Al describir el color de un objeto, no lo hacemos en términos de sus colores primarios. Es difícil trabajar con el modelo RGB porque no se relaciona con la manera en la que percibimos el color: moverse en una dirección dada en el cubo RGB no necesariamente produce un color que es perceptiblemente consistente con el cambio en cada uno de los canales. Por ejemplo: si empezamos con el blanco y le restamos el azul, obtenemos amarillo. Pero si empezamos con el rojo y le sumamos el azul, obtenemos rosado [4]. . Es por esto que otros modelos se basan en un subespacio más acorde a la percepción humana del color, como los espacios HSV o Lab.

El modelo de colores HSV (hue, saturation, value, por sus siglas en inglés) rearregla la geometría del cubo RGB en un intento de ser más intuitivo y más cercano a la percepción humana del color. En particular, es un subespacio expresado en el sistema de coordenadas polares cilíndricas, denominado el modelo “hexcone”, cuyos primeros dos componentes son el tono y saturación de un color. El tercero es su *valor*, que representa qué tanto “negro” tiene el color (podríamos decir que es el opuesto de la saturación, que mide qué tanto blanco tiene el color) [5].

El sistema HSV, no obstante, tiene varias desventajas. A pesar de que es un sistema visualmente acertado, que describe los colores de manera numérica e intuitiva, no transmite mucho de la complejidad de la apariencia del color. Por

ejemplo, los valores del blanco y azul primario son iguales, aunque en realidad el azul tiene aproximadamente un 10 % de la luminancia del blanco. Cambiar cualquier dimensión en el espacio HSV resulta en cambios no uniformes a las tres dimensiones, distorsionando todas las relaciones de color en la imagen. Esto es, el espacio HSV no es perceptiblemente uniforme.

Teniendo en cuenta entonces la necesidad de un modelo de color perceptiblemente uniforme, el modelo de color Lab tiene la intención de permitir que un cambio dado en el color de una imagen resulte en un cambio percetiblemente igual. Lab es el nombre que se le da a dos espacios de color diferentes: CIE 1976 L*a*b y Hunter L,a,b. Ambos espacios se derivan del espacio CIE XYZ; la diferencia entre ellos es que CIELAB se calcula con la raíz cúbica del espacio XYZ y Hunter Lab utilizando con la raíz cuadrada [6]. El más popular es CIE 1976 L*a*b. Así, el modelo Lab se compone de un subespacio tridimensional real, en el cual el canal L denota la claridad como negra cuando $L = 0$ y blanca cuando $L = 100$. Los canales *a* y *b* denotan los ejes verde-rojo y azul-amarillo respectivamente, de tal manera que los valores más negativos de los ejes *a* y *b* son verde y azul y los valores más positivos son rojo y amarillo [7]. El espacio de color Lab incluye todos los colores en el espectro, incluso aquellos que no son perceptibles por el ojo humano. No sólo eso, sino que los colores en este modelo son independientes del mecanismo por el cual se creen o se muestren [8].

Es por todo lo anterior, entonces, que el mejor espacio de color con el que trabajar es el Lab. De esta manera, nuestro descriptor para cada píxel en una imagen es un vector con los valores del píxel en los canales L, a y b de la imagen. Sin embargo, aún podemos mejorar nuestro descriptor si incluimos descriptores para las coordenadas de cada píxel (normalizadas), de tal manera que se refuerzan no sólo relaciones entre colores similares sino también relaciones espaciales entre píxeles [1].

2.. Metodología de pruebas

La base de datos BSDS500, disponible en [3], se compone de 500 imágenes naturales, divididas en un set de entrenamiento y un set de test: en el set de entrenamiento hay 300 imágenes, y en el de test 200. Todas las imágenes están a color, en formato *.jpg* Para cada una de las imágenes hay en promedio 5 sets de anotaciones manuales hechas por humanos, de tal manera que el benchmark compara las segmentaciones hechas por kmeans y GMM con las anotaciones para cada imagen y genera una curva Precision-Recall donde

$$P = \frac{TP}{TP + FP}$$

y

$$R = \frac{TP}{TP + FN}$$

Es decir, la precisión mide la relevancia de los resultados, y la cobertura mide la fracción de instancias relevantes obtenidas. Un sistema con alta cobertura pero poca precisión obtiene muchos verdaderos positivos, pero la mayoría de las etiquetas son falsas en comparación a las etiquetas de anotación. De manera similar, un sistema con alta precisión pero poca cobertura obtiene pocos resultados, pero sus etiquetas son correctas. Es por esto que lo deseable en un algoritmo de segmentación es obtener un punto en el que ambos valores sean máximos.

3.. Resultados

A continuación se presentan tres imágenes y sets de segmentaciones según GMM o Kmeans: a cada una se le aplicó Kmeans y GMM para 5 diferentes números de clusters (2,4,5,6,8 clusters).

Figura 1. Imágenes ejemplo

Figura 2. Clustering Kmeans imagen edificios

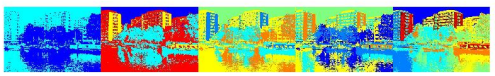

Figura 3. Clustering GMM imagen edificios

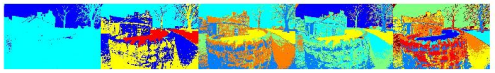

Figura 4. Clustering Kmeans imagen casa

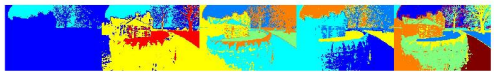

Figura 5. Clustering GMM imagen casa

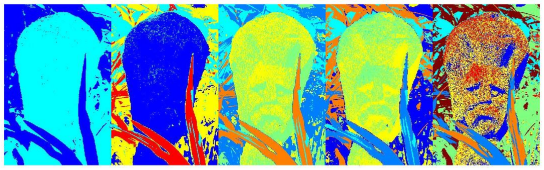

Figura 6. Clustering Kmeans imagen tótem

Figura 7. Clustering GMM imagen tótem

En las anteriores imágenes, es claro un patrón a medida que aumenta el número de clusters. Aparecen más elementos y más complejidad en la imagen a medida que se intenta segmentar la imagen en regiones adicionales que posiblemente sean uniformes. Adicionalmente, las regiones segmentadas difieren drásticamente si se hacen por Kmeans o GMM, sobre todo al agrupar un alto número de clusters, como se puede ver en la última imagen a la derecha en las figuras 2 y 3, 4 y 5 y 6 y 7. Ambos métodos presentan graves dificultades para agrupar textura (ventanas de los edificios en figuras 2 y 3, textura de la piedra en figuras 6 y 7), dividiendo regiones uniformes en muchos clusters de tal manera que se pierde el sentido visual de la imagen y no se entiende muy bien qué corresponde a qué.

A continuación se presenta la curva P-R obtenida con los métodos implementados.

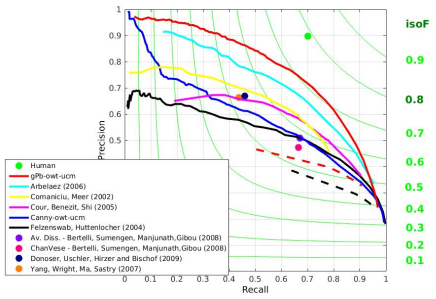

Figura 8. Curvas de Precision-Recall para varios métodos. Las líneas punteadas son GMM (rojo) y Kmeans (negro)

En esta gráfica se puede observar que ambos métodos tuvieron baja precisión y baja cobertura en general. En particular, a medida que aumenta la precisión, disminuye la cobertura, y viceversa, demostrando cómo una mejora en una medida afecta negativamente la otra: aumentar la cobertura implica aumentar las posibilidades de detectar falsos positivos, mientras que aumentar la precisión disminuye las posibilidades de detectar todos los verdaderos positivos.

4.. Discusión

En general, ambos algoritmos son pobres en comparación con los otros. Esto indica que los grupos o segmentos en la imagen no necesariamente se pueden dividir correcta o precisamente según los descriptores que utilizamos. Efectivamente, para agrupar textura necesitamos descriptores más sofisticados como las respuestas a un banco de filtros: para segmentar correctamente la imagen según sus elementos, el color y la posición no son descriptores suficientemente sofisticados para lograr esta tarea de manera confiable, sobre todo en imágenes complejas que presentan texturas y muchos elementos diferentes que pueden tener colores similares y estar relativamente cercanos. Adicionalmente, la segmentación depende completamente del número de clusters especificados previamente: si se especifican muy pocos, no se encuentran todos las regiones en la imagen, y si se especifican demasiados, la imagen se sobresegmenta.

No obstante lo anterior, el método de GMM funciona mejor que Kmeans. Esto es porque permite una flexibilidad en la forma de los clusters en el espacio de representación, y asigna los datos a cada componente de la mezcla de Gaussianas según la máxima probabilidad posterior de que un dato corresponda a ese componente. En ese sentido, la mezcla de gaussianas tiene menos “ruido” en las segmentaciones y no presenta sobresegmentación tan agudamente como Kmeans.

5.. Mejoras

Para obtener resultados más uniformes y similares a las anotaciones manuales, se deberían utilizar descriptores ade-

cuados para cada imagen. En particular, se podría evaluar si una imagen presenta mucha textura o no, a partir de lo cual se podrían construir descriptores diferentes. En imágenes relativamente sencillas, con poca textura, regiones claramente delimitadas por color y espacio, los algoritmos de segmentación por Kmeans y GMM con los descriptores utilizados pueden funcionar bien. Sin embargo, fracasan gravemente en imágenes con texturas complejas. Es por esto que una decisión sobre qué descriptores utilizar para cada imagen individualmente podría ayudar mucho al desempeño de la segmentación por clustering.

Por otro lado, existen métodos de segmentación por clustering diferentes como Mean Shift. Teniendo en cuenta que estos métodos también detectan centroides, una idea podría ser comparar los centroides entre algoritmos, haciendo un promedio de ellos y reasignando los datos según distancias al centroide o probabilidades de correspondencia.

Referencias

[1] D. Forsyth and J. Ponce, "Computer vision." Prentice Hall, 2012, pp. 252–273.

[2] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423.

[3] Berkeley.edu, "The berkeley segmentation dataset and benchmark." [Online]. Available: https://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

[4] C. Solomon and T. Breckon, "Fundamentals of digital image processing." Wiley-Blackwell, 2011, pp. 86–107.

[5] A. R. Smith, "Color gamut transform pairs," *ACM SIGGRAPH Computer Graphics*, vol. 12, no. 3, pp. 12–19, 1978.

[6] *Measuring color using Hunter L,a,b vs. CIE 1976 L*a*b*. [Online]. Available: http://www.hunterlab.com/an-1005b.pdf?r=false

[7] *Explanation of the LAB color space*. [Online]. Available: http://www.aces.edu/dept/fisheries/education/pond_to_plate/documents/ExplanationoftheLABColorSpace.pdf

[8] Mathworks.com, "Lab color - matlab." [Online]. Available: http://www.mathworks.com/discovery/lab-color.html?refresh=true