

Gaussian Mixture Model, K-Means y Density Based Scan: Segmentación de tumores cerebrales.

1st Fabrizio Franco
Ciencia de la Computación
Universidad de Ingeniería y Tecnología
Lima, Perú
fabrizio.franco@utec.edu.pe

2nd Jeremy Matos Cangalaya
Ciencia de la Computación
Universidad de Ingeniería y Tecnología
Lima, Perú
jeremy.matos@utec.edu.pe

Index Terms—Inteligencia Artificial, Agrupamiento, K-Means, Density Based Scan (DBScan), Gaussian Mixture Model (GMM).

I. INTRODUCCIÓN

Este artículo propone una solución al problema de identificación de tumores en tomografías cerebrales de humanos. Para ello, se plantea la utilización de tres algoritmos de *clusterización*: *K-Means*, *Density Based Scan (DBScan)* y *Gaussian Mixture Model (GMM)*. La elección de hiperparámetros para cada algoritmo ha sido explorada experimentalmente. Así mismo, se evaluarán los mejores y peores resultados obtenidos de cada algoritmo, tanto para la segmentación de tumores como para la búsqueda por similitud.

La figura 2 muestra el flujo de trabajo que se realiza en este proyecto. En primer lugar, las tomografías son convertidas a un tamaño fijo de imagen: $100 \cdot 100$ pixels. Luego, cada una de las imágenes es segmentada a través de los algoritmos de *clusterización*. Posteriormente, se procede a extraer el *cluster* correspondiente al tumor, guardarlo en una imagen y aplicarle una transformación *Wavelet*. Finalmente, se consulta los N tumores más similares. Cada paso del proceso será profundizado en la sección de Experimentación.

II. BASE DE DATOS

La base de datos está conformada por aproximadamente 300 imágenes de diferentes tamaños. Cada imagen representa una tomografía cerebral, la cual puede, o no, contener regiones tumorales. Además, no todas las imágenes están tomadas desde el mismo ángulo; sin embargo, la solución propuesta es capaz de funcionar correctamente a pesar de ello.

Fig. 1: Ejemplo de tomografía

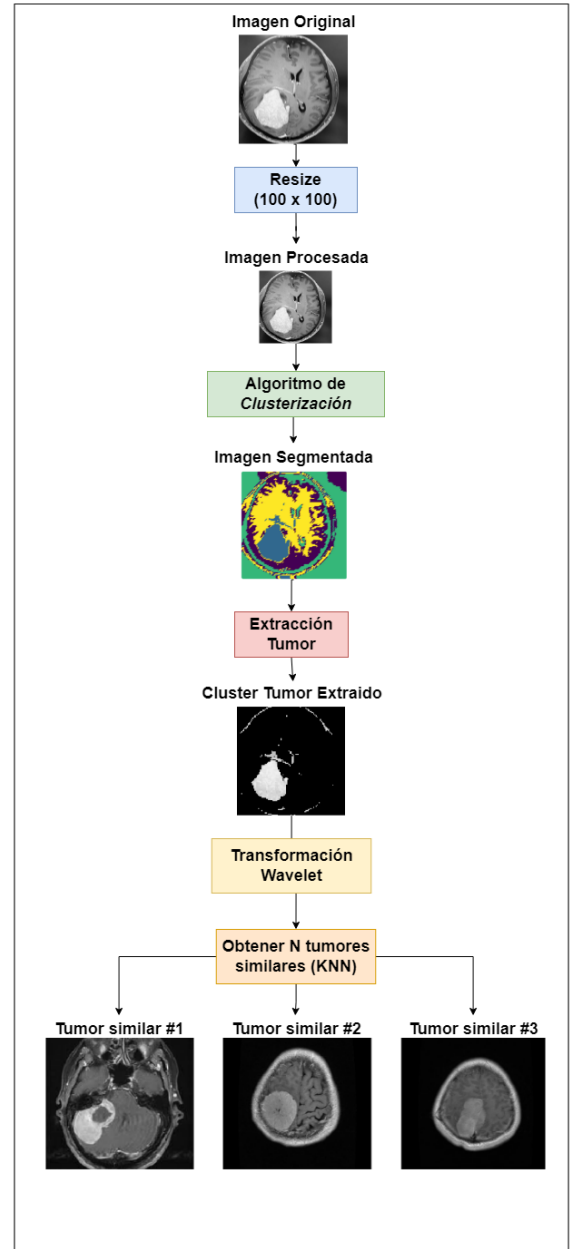


Fig. 2: Esquema de trabajo

A. Representación de imágenes

Cada imagen, antes de ser utilizada como *input* de los algoritmos de *clusterización*, ha sido reescalada a un tamaño fijo de 100x100 pixels. Cada pixel esta representado como un arreglo de elementos [R, G, B]. Por lo que cada imagen resulta en una matriz de dimensiones (100, 100, 3). La figura 3 muestra las 3 capas de color de una misma tomografía cerebral.

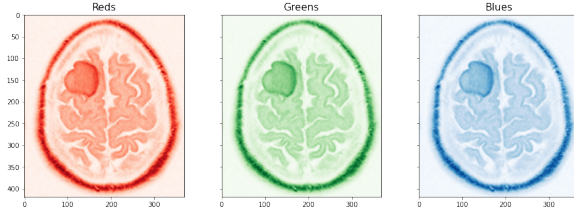


Fig. 3: capas RGB

III. EXPLICACIÓN DE LOS MODELOS

Agrupamiento en inteligencia artificial, hace referencia a una familia de algoritmos no supervisados cuyo objetivo es descubrir patrones o agrupaciones intrínsecas entre los datos no etiquetados.

A. K-Means

Es el algoritmo de agrupamiento basado en centroides más popular, debido a su facilidad de implementación. Tiene como objetivo encontrar los K mejores centroides que segmenten a los datos. En este algoritmo, un dato pertenece al k-ésimo centroide sí y solo sí es su centroide más cercano. La figura 4 representa gráficamente lo antes mencionado.

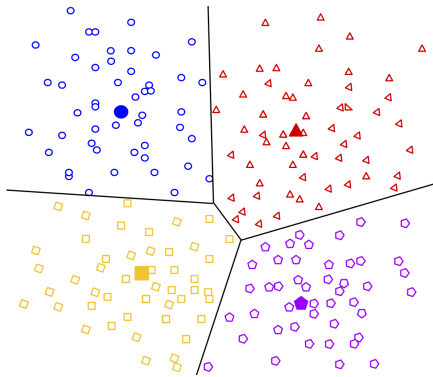


Fig. 4: K-means

Ventajas:

- Su implementación es sencilla.
- Escala a grandes conjuntos de datos

Desventajas:

- Al ser un algoritmo basado en centroides es muy sensible a las condiciones iniciales.
- Los *outliers* no son identificados por el algoritmo y son tratados con igual relevancia.

- Al ser un algoritmo soportado en una función de distancia, de acuerdo al número de dimensiones del *dataset*, esta puede no ser significativa y afectar los resultados.

Input: D: datos

Output: K: centroides

$K \leftarrow \text{inicializar_centroides}()$

$K_{aux} \leftarrow \text{lista_vacía}()$

while K_{aux} diferente a K **do**

$K_{aux} \leftarrow K$

$K \leftarrow \text{actualizar_centroides}()$

end

Algoritmo 1: K-Means

1) *Elección de parámetros:* Este algoritmo solo toma un hiper parámetro, la cantidad de centroides. Para elegir la cantidad adecuada de centroides se utilizó *Elbow Method*. Es decir, elegir el punto de inflexión a partir del cual, el error ya no se reduce lo suficiente al aumentar más centroides. El error en la figura 5 está dado por *Within-Cluster-Sum of Squared Errors (WSS)*:

$$WSS = \sum_{i=1}^{N_C} \sum_{x \in C_i} (x - C_i)^2 \quad (1)$$

Donde N_C es la cantidad de centroides y C_i es cada centroide.

Este método de elección de parámetros se aplico a múltiples imágenes y, como resultado, se obtuvo que **K=4 es la cantidad adecuada de centroides**, por lo que la experimentación de *K-means* fue llevada a cabo con dicho valor. En la figura 6 se muestra con ejemplo del buen desempeño obtenido.

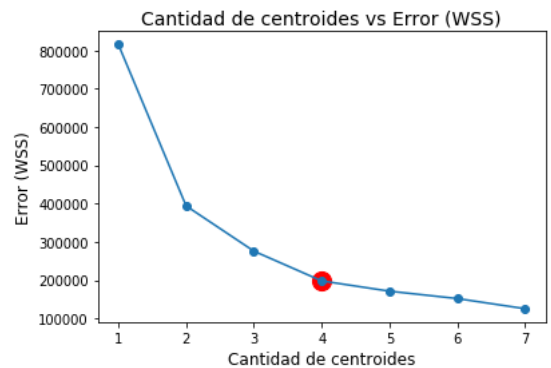


Fig. 5: *Elbow Method*

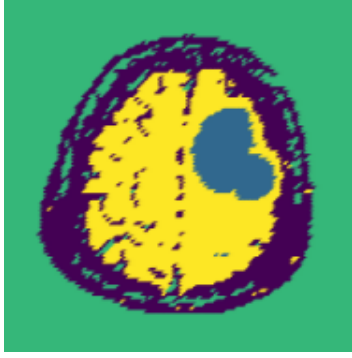


Fig. 6: Ejemplo con $K=4$

2) Criterios de inicialización de centroides:

- Aleatorio: Como su nombre lo indica, elige K muestras del *dataset* como centroides iniciales. Este criterio tiene alta volatilidad y su desventaja es que el resultado del algoritmo puede ser una cantidad de centroides menor a la esperada, ya que se solaparon en el proceso.
- *K-Means++*: Escoge algorítmicamente K centroides de modo que estén lo más lejos entre sí. Lo logra escogiendo un primer centroide aleatoriamente y los centroides restantes se generan de acuerdo a una distribución de probabilidad proporcional a la distancia al cuadrado de cada dato a su centroide más cercano.

Para la fase de experimentación de este algoritmo se utilizó el criterio de inicialización de centroides *K-means++*.

B. Density Based Scan (DBScan)

Es un algoritmo de agrupamiento que localiza regiones de alta densidad y las separa de las regiones con baja densidad para formar los grupos. Además, etiqueta los datos que no pertenecen a ningún grupo como ruido. La figura 7 muestra el ruido del *dataset* en color rojo. Se soporta en 2 hiperparámetros cuyos valores afectan significativamente su comportamiento: epsilon o radio (ϵ) y un umbral de densidad o *threshold* (thr).

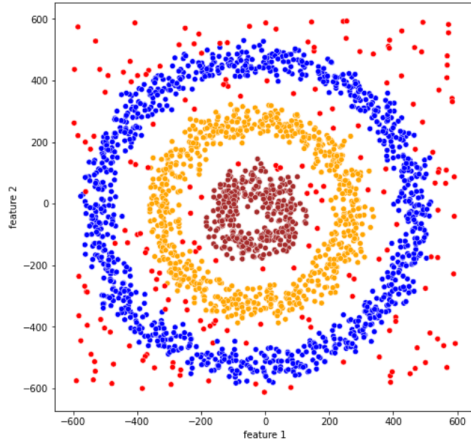


Fig. 7: DBScan

Ventajas:

- Es robusto frente a los *outliers* y al ruido del *dataset*.
- Puede generar grupos de diferentes tamaños y formas.
- No requiere un número específico de clusters

Desventajas:

- Es altamente sensible a sus hiperparámetros.
- No puede agrupar bien conjuntos de datos con alta variación.

Hiperparámetros:

- Epsilon/Radio: limita el radio de búsqueda para posteriormente evaluar si se puede formar un nuevo cluster.
- *Threshold*/Umbral: define la cantidad límite de elementos que puede generar un nuevo cluster.

Input: D : datos
Input: ϵ : radio de búsqueda
Input: t : umbral de densidad
 $\text{etiqueta} \leftarrow \text{inicializar_etiquetas}()$
for dato **en** D **do**
 if $\text{etiqueta}[\text{dato}] \neq \text{INDEFINIDO}$ **then**
 | **continue**;
 end
 $N \leftarrow \text{obtener_cercanos}(\text{dato}, D, t, \epsilon)$
 if $N.\text{longitud} < t$ **then**
 | $\text{etiqueta}[\text{dato}] = \text{RUIDO}$
 end
 $c \leftarrow \text{obtener_etiqueta_nuevo_cluster}()$
 $\text{etiqueta}[\text{dato}] \leftarrow c$
 $S \leftarrow N \setminus \text{dato}$
 for q **en** S **do**
 if $\text{etiqueta}[q] == \text{RUIDO}$ **then**
 | $\text{etiqueta}[q] \leftarrow c$
 end
 if $\text{etiqueta}[q] \neq \text{INDEFINIDO}$ **then**
 | **continue**;
 end
 $\text{etiqueta}[q] \leftarrow c$
 $NN \leftarrow \text{obtener_cercanos}(q, D, t, \epsilon)$
 if $NN.\text{longitud} < t$ **then**
 | **continue**;
 end
 $S \leftarrow S \cup NN$
 end
end

Algoritmo 2: DBScan

El algoritmo itera sobre todos los puntos, establece un conjunto de *clusters* iniciales y va adicionando todos los puntos dentro del radio de búsqueda (Expansión). Luego, de ser necesario, crea un *cluster* nuevo. Finalmente, los puntos sin un *cluster* asignado son considerados como ruido.

1) *Elección de parámetros*: Para este algoritmo se utilizó *Cross-Validation* con el fin de seleccionar los mejores valores para epsilon (ϵ) y para el umbral de densidad (thr). Los valores que se probaron en las combinaciones fueron $\epsilon = [0.1, 0.5, 1, 2, 5]$ y $\text{thr} = [5, 10, 25, 75, 100]$. Las combinaciones más resaltantes se encuentran en la tabla I. La combinación de hiper-parámetros A muestra buenos resultados para segmentar el tumor. Aunque las regiones no tumorales no están separadas en diferentes componentes. Por otro lado, la combinación B es la de mejor desempeño, sin embargo, su tiempo de ejecución fue 15 veces mayor en promedio. Y, para efectos, de la solución no resulta beneficioso la correcta segmentación entre regiones no tumorales, pues serán descartadas para los siguientes pasos. Finalmente, las combinaciones C y D forman parte de un largo conjunto de combinaciones infructíferas que no lograron una segmentación adecuada de la región tumoral. **Por lo mencionado, se seleccionó la combinación A de parámetros como la más adecuada.**

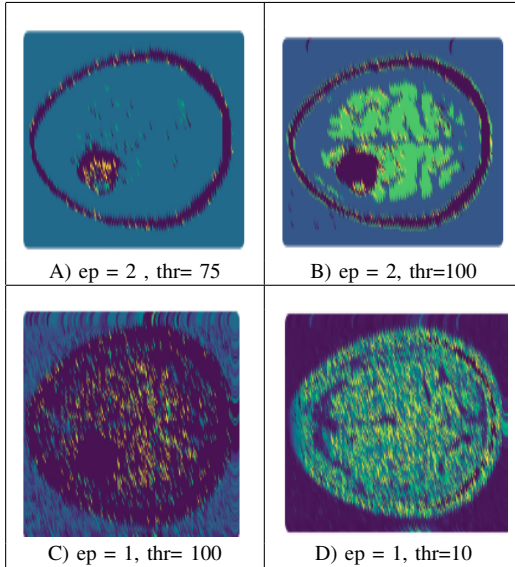


Fig. 9: Campanas de GMM

Ventajas:

- Respecto a otros algoritmos de agrupamiento, este tiene en cuenta la varianza de los datos.
- A diferencia del *K-means*, el cual realiza una clasificación estricta, *GMM* realiza una clasificación suave, proporcionando las probabilidades de que un elemento del *dataset* pertenezca a cada uno de los posibles grupos (representados por las distribuciones).

Desventajas:

- Ya que se soporta en un *K-means*, este algoritmo tiene una mayor complejidad computacional, lo que se traduce en que su tiempo de convergencia es mayor.

El desafío que presenta el *GMM* es ubicar las funciones de distribución para obtener la máxima probabilidad conjunta, conocido también como máxima verosimilitud (*likelihood*).

Input: K: clusters

Output: N: iteraciones

$\text{Mu, Cov, Pi} \leftarrow \text{inicializar_aleatoriamente}()$

for $i \leftarrow N$ **do**

$\gamma \leftarrow \text{Expectation_step}(\text{Mu, Cov, Pi})$

$\text{Mu, Cov, Pi} \leftarrow \text{Maximization_step}()$

end

Algoritmo 3: GMM

GMM usa el algoritmo EM (*Expectation Maximization*) para lograr obtener la máxima verosimilitud.

El algoritmo *Expectation Maximization* consta de 2 pasos definidos:

- E_step: Evalúa la responsabilidad dado los valores actuales de Mu , Cov y Pi .
- M_step: Reestima los parámetros usando el valor de responsabilidad del paso previo.

1) *Elección de parámetros*: Para elegir la cantidad de correcta de componentes y el tipo de *clusters* iniciales se aplicó *Silhouette Score*. Dicho indicador se utiliza para medir la calidad de segmentación del modelo. Es un valor que oscila entre -1 y 1. Calcula que tan separados se encuentran los componentes. Esta dado por:

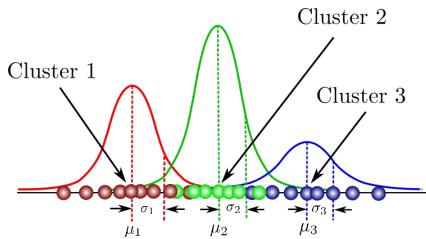


Fig. 8: GMM

C. Gaussian Mixture Model (GMM)

Mezcla Gaussiana es un modelo probabilístico que busca ubicar una cantidad K de funciones de distribución gaussiana en la dimensión de un *dataset* de modo que cada distribución sea considerada como un *cluster* o componente. De este modo, un elemento X_i del *dataset* pertenece al *cluster* K_j para el cual su probabilidad es máxima.

$$Silhouette = \sum_{i=1}^N \frac{(b_i - a_i)}{\max(a_i, b_i) * N} \quad (2)$$

Donde N es la cantidad de datos. Cada a_i representa la media de la distancia entre el dato y todos los demás datos en el mismo componente. Y cada b_i representa la media de la distancia entre el dato y todos los demás datos pertenecientes al componente más cercano. De tal manera que mientras más cercano sea el valor a 1, significa que los componentes se encuentran más separados y son densos. Un valor cercano a 0 indicaría datos dispersos. Y, valores negativos indican segmentación errónea. Para elegir la mejor combinación de parámetros se corrieron 20 iteraciones por combinación. La figura 10 muestra los resultados obtenidos. Donde se observa que utilizar centroides de *K-means* como punto de partida mejora considerablemente los resultados. Y, a su vez, cuando se modela con 5 componentes se maximiza el *Silhouette score*.

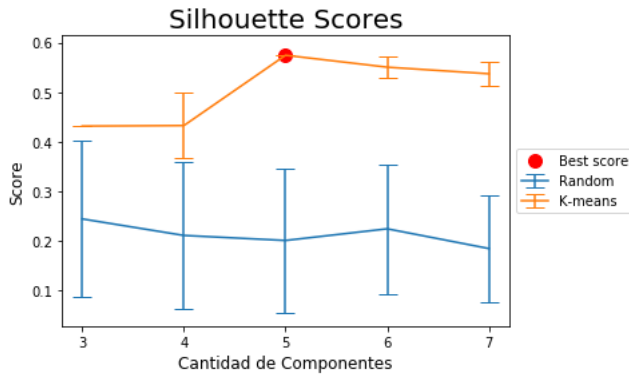


Fig. 10: Silhouette Scores

Por lo analizado previamente, se eligió K-Means y 5 componentes como la combinación adecuada de hiperparámetros para Gaussian Mixture Model en este conjunto de datos.

La implementación presentada en GMM utiliza la librería Sklearn de acuerdo a coordinación con el docente.

IV. EXPERIMENTACIÓN

Para la ejecución de los modelos, se realizó un enfoque paralelo mediante hilos físicos. Se segmenta los datos en H subsets de igual tamaño, donde H es la cantidad de hilos físicos de la computadora. Para ello se crea un proceso independiente para cada subset. Por lo que se consiguió reducir el tiempo de ejecución en una escala H veces más rápido.

En el repositorio de GitHub (Anexo 1) se encuentra de forma completa los mejores y peores resultados de la experimentación. A continuación, se explicará un subgrupo de dichos resultados de forma concisa.

A. Segmentación y extracción de secciones tumorales

1) *K-Means*: En las tablas II y III se presentan los resultados obtenidos. Se puede observar que hay casos de excelente rendimiento, así como también algunos donde el tumor no logra ser aislado del cerebro debido a su similitud con las regiones no tumorales. También es posible notar que en algunos casos, la corteza cerebral es agrupada con el tumor.

• Mejores resultados

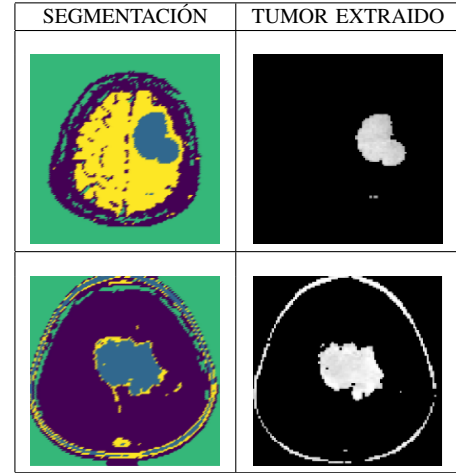


TABLE II: Mejores resultados *K-Means*

• Peores resultados

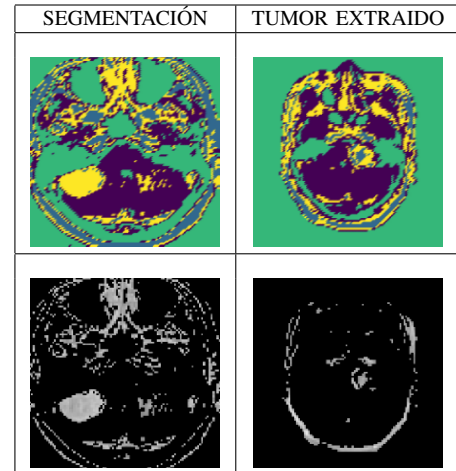


TABLE III: Peores resultados *K-Means*

2) *Density Based Scan (DBScan)*: En las tablas IV y V se presentan los resultados obtenidos. Se puede observar que hay casos de excelente rendimiento, así como también algunos donde el tumor es considerado como ruido debido a que dicha tomografía cuenta con píxeles muy similares entre sí. También es posible notar que en todos los casos la corteza cerebral es segmentada junto al tumor.

• Mejores resultados

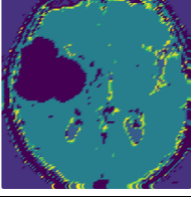
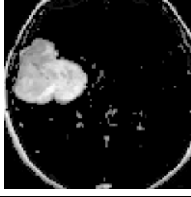
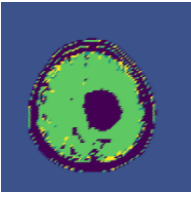
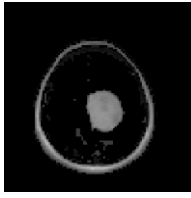
SEGMENTACIÓN	TUMOR EXTRAÍDO
	
	

TABLE IV: Mejores resultados *DBScan*

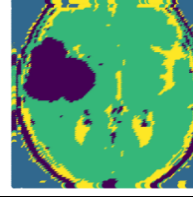
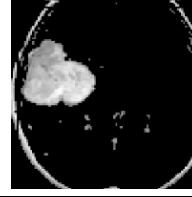
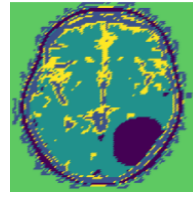
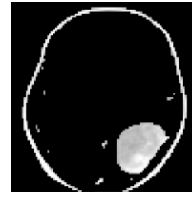
SEGMENTACIÓN	TUMOR EXTRAÍDO
	
	

TABLE VI: Mejores resultados *GMM*

- Peores resultados

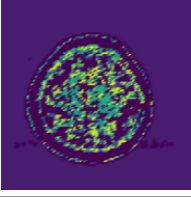
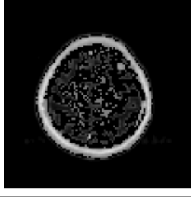
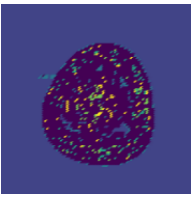
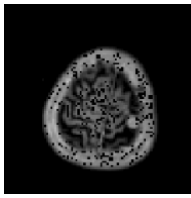
SEGMENTACIÓN	TUMOR EXTRAÍDO
	
	

TABLE V: Peores resultados *DBScan*

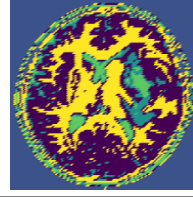

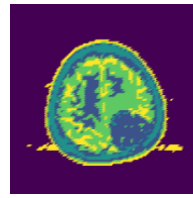

SEGMENTACIÓN	TUMOR EXTRAÍDO
	
	

TABLE VII: Peores resultados *GMM*

3) *Gaussian Mixture Model (GMM)*: En las tablas VI y VII se presentan los resultados obtenidos. Se puede observar que hay casos de excelente rendimiento; sin embargo también existen casos donde el tumor no es separado de regiones cerebrales no tumorales similares, principalmente la región central del cerebro. Dicho fenómeno se debe a la naturaleza de la data. También es posible notar que en todos los casos la corteza cerebral es segmentada junto al tumor.

- Mejores resultados

- Peores resultados

B. Transformación Wavelet

Se utilizó una transformación Wavelet para representar en un vector K dimensional cada tumor extraído tras la segmentación para un *Dataset* de tamaño D . La cantidad de cortes fue elegida experimentalmente. Como se observa en la figura 11, entre 1 y 3 cortes la cantidad de características supera el tamaño del conjunto de datos, por lo que no es recomendable. Entre 3 y 5 cortes la cantidad de características es menor al tamaño del conjunto de datos. Y adicionalmente, se obtuvieron buenos resultados de similitud al realizar *KNN* sobre un *KD-Tree*. Finalmente, de 6 cortes en adelante, la calidad de los resultados disminuye drásticamente, ya que se pierde la información referente a la posición del tumor en el cráneo. Es decir, los tumores recuperados en la búsqueda por similitud ya no se encuentran en posiciones similares del cerebro. **Por lo mencionado, se ha elegido utilizar 5 cortes en la transformación Wavelet, puesto que se obtiene resultados adecuados con la menor cantidad de características posibles.**

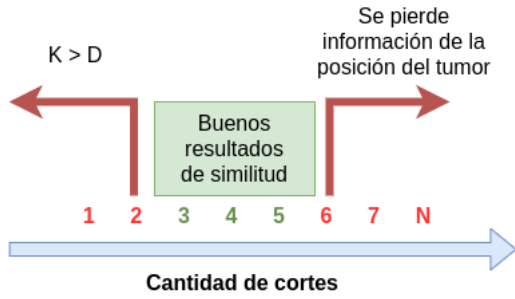


Fig. 11: Experimentación Wavelet

C. Búsqueda por similitud

Para la búsqueda por similitud se utilizó el algoritmo K vecinos más próximos. Para ello, se tomó como estructura de apoyo un KD -Tree, por lo que se asegura tiempo de complejidad $N \cdot \lg(N)$.

1) *K-Means*: Las tabla VIII muestra resultados donde se puede apreciar la similitud en color, ubicación y tamaño de los tumores, mientras que la tabla IX muestra resultados malos provocados por imágenes cuyo tumor no ha sido segmentado de la mejor forma, indexando como parecidas imágenes disímiles.

- Mejores resultados

TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2

TABLE VIII: Mejores resultados KNN aplicado a *K-Means*

- Peores resultados

TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2

TABLE IX: Peores resultados KNN aplicado a *K-Means*

2) *Density Based Scan (DBScan)*: Se puede apreciar en la tabla X resultados ligeramente mejores que los de *K-Means* ya que *DBScan* segmentó tanto el cráneo como el tumor, por tener colores similares en la imagen original, añadiendo características que harían efecto en el resultado de aplicar *Wavelet*. Sin embargo, esto también tuvo efecto en los peores resultados presentados en la tabla XI, ya que devuelve imágenes con forma y tamaño de los cráneos similares, pero no de los tumores.

- Mejores resultados

TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2

TABLE X: Mejores resultados KNN aplicado a *DBScan*

- Peores resultados

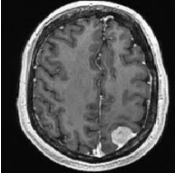
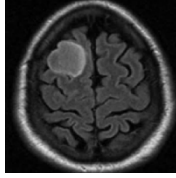
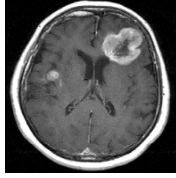
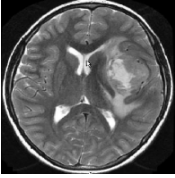
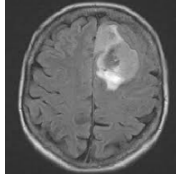
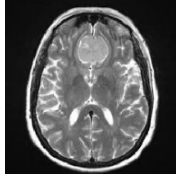
TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2
		
		

TABLE XI: Peores resultados *KNN* aplicado a *DBScan*

3) *Gaussian Mixture Model (GMM)*: Se espera que *GMM* al ejecutarse sobre un *K-Means*, de resultados más precisos, segmentando exclusivamente el tumor de cada imagen y devolviendo imágenes bastante similares como lo presenta la tabla XII. Sin embargo, como no se segmenta el cráneo se pierde información de la posición relativa. Además, para imágenes donde el tumor no se diferencia de sus píxeles próximos *GMM* lo incluye en un grupo al cual no le corresponde, provocando resultados como los presentados en la tabla XIII

- Mejores resultados

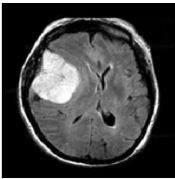
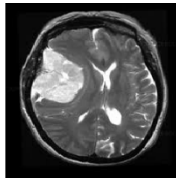
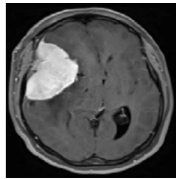
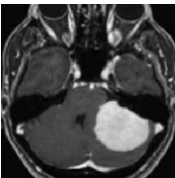
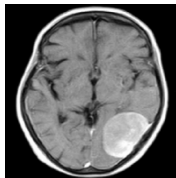
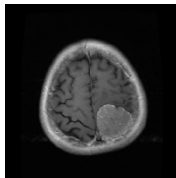
TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2
		
		

TABLE XII: Mejores resultados *KNN* aplicado a *GMM*

- Peores resultados

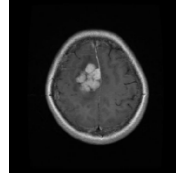
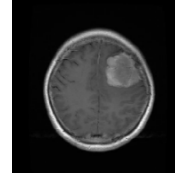
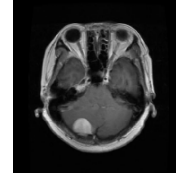
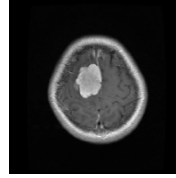
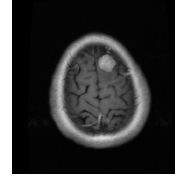
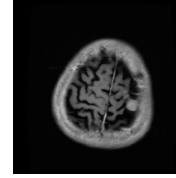
TUMOR ORIGINAL	TUMOR SIMILAR #1	TUMOR SIMILAR #2
		
		

TABLE XIII: Peores resultados *KNN* aplicado a *GMM*

V. CONCLUSIONES

- 1) Para asegurar el buen rendimiento de los algoritmos, se utilizaron diversos métodos para la elección óptima de sus hiper parámetros y esto a su vez minimiza el error en fases posteriores del proceso de experimentación. Se recomienda utilizar este enfoque de elección de parámetros basado en métricas en proyectos similares, con la consideración de que cada tipo de modelo contempla métricas diferentes para la elección de sus parámetros.
- 2) Los 3 algoritmos muestran un buen desempeño en gran parte de los datos, por lo que se concluye que es posible aplicar algoritmos de agrupamiento basados en densidad, centroides y distribución a este *dataset* siempre que la elección de parámetros esté fundamentada.
- 3) Agrupar el tumor junto a la corteza cerebral permite conocer la posición relativa del tumor respecto al cráneo. Y, esto conlleva que los resultados por similitud entreguen resultados de tumores no solo similares en forma, sino también en posiciones lo más cercanas al tumor original que se está consultado.
- 4) Utilizar el valor correcto de cortes para la transformación Wavelet permite reducir la dimensionalidad de la representación tumoral sin afectar el rendimiento. Muchos cortes sobre los datos pueden producir under-fitting, mientras que pocos cortes puede producir over-fitting. El cálculo de los cortes requiere necesariamente un proceso experimental para determinar dicho valor.
- 5) Utilizar un enfoque paralelo en la experimentación fue fundamental para el desarrollo del proyecto en los plazos establecidos. Se recomienda mantener este formato para proyectos similares.

REFERENCES

- [1] Scikit. (2022). scikit-learn: machine learning in Python — scikit-learn 1.0.2 documentation. Scikit Learn. Retrieved 2022, from <https://scikit-learn.org/stable/>
- [2] Saul Dobilas (2021, May 21). GMM: Gaussian Mixture Models — How to Successfully Use It to Cluster Your Data? Medium. <https://towardsdatascience.com/gmm-gaussian-mixture-models-how-to-successfully-use-it-to-cluster-your-data-891dc8ac058f>

- [3] Mayo, Matthew (2022, 13 May) Centroid Initialization Methods for k-means Clustering. KDNuggets. Retrieved 2022, from <https://www.kdnuggets.com/2020/06/centroid-initialization-k-means-clustering.html>
- [4] Foley, Daniel (2019, 8 Mar) Gaussian Mixture Modelling (GMM). Towards Data Science. Retrieved from: <https://towardsdatascience.com/gaussian-mixture-modelling-gmm-833c88587c7f>
- [5] VIPUL GANDHI (2022, May 20). Gaussian Mixture Models Clustering - Explained. Retrieved 2022 from Kaggle. <https://www.kaggle.com/code/vipulgandhi/gaussian-mixture-models-clustering-explained/notebook>

VI. ANEXOS

- [Enlace al repositorio de Github](#)