EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Centro Nacional de Investigación y Desarrollo Tecnológico

Tesis de Doctorado

Mejora del algoritmo de agrupamiento Fuzzy C-Means

presentada por

MC. Sandra Silvia Roblero Aguilar

como requisito para la obtención del grado de

Doctora en Ciencias de la Computación

Director de tesis

Dr. Joaquín Pérez Ortega

Codirector de tesis

Dr. José Crispín Zavala Díaz

Cuernavaca, Morelos, México. Mayo de 2023.

cenidet
Centro Nacional de Investigación y Desarrollo Tecnológico

ESC\FORDOC09

Cuernavaca, Morelos, 22/mayo/2023

ASUNTO: ACEPTACIÓN DEL TRABAJO DE TESIS DOCTORAL

MARÍA YASMIN HERNÁNDEZ PEREZ
JEFA DEL DEPARTAMENTO DE CIENCIAS COMPUTACIONALES
PRESENTE

Los abajo firmantes, miembros del Comité Tutorial de la Tesis Doctoral de la alumna **SANDRA SILVIA ROBLERO AGUILAR** manifiestan que después de haber revisado su trabajo de tesis doctoral titulado **"MEJORA DEL ALGORITMO DE AGRUPAMIENTO FUZZY C-MEANS"**, realizado bajo la dirección de Joaquín Pérez Ortega, el trabajo se ACEPTA para proceder a su impresión.

ATENTAMENTE
"Excelencia en Educación Tecnológica®
"Educación Tecnológica al Servicio de México"

JOAQUÍN PÉREZ ORTEGA
CENIDET

ALICIA MARTÍNEZ REBOLLAR
CENIDET

JAVIER ORTIZ HERNÁNDEZ
CENIDET

JOSÉ CRISPÍN ZAVALA DÍAZ

MARÍA YASMIN HERNÁNDEZ PEREZ
CENIDET

ADRIANA MEXICANO SANTOYO
TECNM/CAMPUS CIUDAD VICTORIA

EDUCACIÓN

23 MAY 2023

CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
SERVICIOS ESCOLARES
RECIBIDO

C.c.p. Lic. Silvia del Carmen Ortiz Fuentes / Jefa del Depto. de Servicios Escolares
Dr. Carlos Manuel Astorga Zaragoza / Subdirector Académico
Expediente

Cuernavaca, Mor., 23/mayo/2023
No. De Oficio: SAC/077/2023
Asunto: Autorización de impresión de tesis

SANDRA SILVIA ROBLERO AGUILAR
CANDIDATA AL GRADO DE DOCTORA EN CIENCIAS DE LA COMPUTACIÓN
P R E S E N T E

Por este conducto, tengo el agrado de comunicarle que el Comité Tutorial asignado a su trabajo de tesis titulado **"MEJORA DEL ALGORITMO DE AGRUPAMIENTO FUZZY C-MEANS"**, ha informado a esta Subdirección Académica, que están de acuerdo con el trabajo presentado. Por lo anterior, se le autoriza a que proceda con la impresión definitiva de su trabajo de tesis.

Esperando que el logro del mismo sea acorde con sus aspiraciones profesionales, reciba un cordial saludo.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
"Conocimiento y tecnología al servicio de México"

EDUCACIÓN
CENTRO NACIONAL DE INVESTIGACIÓN Y DESARROLLO TECNOLÓGICO
SUBDIRECCIÓN ACADÉMICA

CARLOS MANUEL ASTORGA ZARAGOZA
SUBDIRECTOR ACADÉMICO

C. c. p. Departamento de Ciencias Computacionales
Departamento de Servicios Escolares

CMAZ/lmz

Dedicatorias

Dedico esta investigación a Dios, que diariamente me llena de muchas bendiciones.

A mi nena preciosa Abril y a mi papi chulo Adrián, por estar siempre conmigo y darme la oportunidad de ser mejor persona.

A mi esposo Oscar, gracias amor mío por coincidir y formar parte de esta pequeña, pero gran familia.

A mi madre Teresita, por todo su apoyo y comprensión, por ser la mejor madre que Dios me dio, por todo su sacrificio, para que sus hijos se realizaran personal y profesionalmente.

A mis hermanos Magdalena, Omar y Sonia, por su cariño incondicional y por estar siempre en los momentos que más los he necesitado.

A José Manuel, Rodrigo y Arturo Rojas, por el gran cariño y cuidados que nos han brindado siempre.

Agradecimientos

Al Tecnológico Nacional de México y en especial al Instituto Tecnológico de Tlalnepantla, mi alma mater, por todas las facilidades brindadas, para que lograra culminar este gran sueño.

Al Centro Nacional de Investigación y Desarrollo Tecnológico, por la formación recibida durante mis estudios de doctorado.

A mi director de tesis, Dr. Joaquín Pérez Ortega, por compartir sus conocimientos, dedicación y ser una gran ser humano.

A mi codirector de tesis, Dr. José Crispín Zavala Díaz y los miembros del comité tutorial: Dra. Alicia Martínez Rebollar, Dr. Javier Ortíz Hernández, Dra. Adriana Mexicano Santoyo, Dra. Yasmín Hernández Pérez, por sus atinadas observaciones y sugerencias.

Un especial agradecimiento a la Dra. Leticia Sánchez Lima por todas sus enseñanzas y paciencia en los seminarios para cada uno de los integrantes del equipo de trabajo.

A mis queridos compañeros de esta travesía, en especial al equipo de trabajo que conforma el Dr. Joaquín. Mil gracias por todo lo que aprendí de ustedes, pero sobre todo por su invaluable amistad.

Resumen

La presente investigación, se ubica en el contexto de los problemas relacionados con las mejoras de los algoritmos de agrupamiento o *clustering*. En particular, el problema que se aborda consiste en reducir el tiempo de procesamiento del algoritmo de agrupamiento *Fuzzy C-Means* (*FCM*) para la solución de grandes *datasets*. El enfoque de solución, consistió en mejorar la fase de inicialización de *FCM* mediante la optimización de la matriz de pertenencia inicial. A este nuevo algoritmo se le denominó *Hybrid OK-Means Fuzzy C-Means* (*HOFCM*).

El problema de agrupamiento de *FCM* es del tipo *NP-Hard*, lo cual justifica el uso de métodos heurísticos para su solución. A la fecha, se han propuesto mejoras en la etapa de inicialización del algoritmo *FCM*. Sin embargo, no están orientadas a la solución de grandes *datasets* como las que se presentan en el *Big Data*. En este sentido, el algoritmo propuesto reduce el número de iteraciones al optimizar los valores iniciales de la matriz de pertenencia. Este enfoque consta de tres pasos: a) generar un conjunto de *n* soluciones de un *dataset* dado, aplicando una variante del algoritmo *K-Means*; b) seleccionar la mejor solución como base para generar la matriz de pertenencia optimizada y c) resolver el *datase*t dado con *FCM*.

Para validar los resultados del algoritmo *HOFCM*, se diseñaron y ejecutaron un conjunto de experimentos compuestos de *datasets* reales y sintéticos. *HOFCM* se contrastó con los algoritmos: *FCM* estándar, *FCM-KMeans*, *FCM++* y *NFCM*. Con base en los resultados experimentales, se observó que *HOFCM* obtuvo una reducción en el tiempo de solución en todos los *datasets* grandes, comparado con el algoritmo *FCM* estándar. Al contrastar los resultados de *HOFCM* con los algoritmos *FCM-KMeans*, *FCM++* y *NFCM*, se observó que, en promedio, fue más rápido 1.51, 2.87 y 3.01 veces respectivamente.

Finalmente, con base en los resultados obtenidos, es posible afirmar que, con esta investigación, se aportaron beneficios para usuarios que buscan resolver el problema de agrupamiento difuso de grandes *datasets*, como los que se presentan en *Big Data* en tiempo razonable.

Abstract

This research is situated in the context of problems related to improvements in clustering algorithms. In particular, the problem addressed consists of reducing the processing time of the *Fuzzy C-Means* (*FCM*) clustering algorithm for the solution of large datasets. The solution approach was to improve the initialization phase of *FCM* by optimizing the initial membership matrix. This new algorithm was called *Hybrid OK-Means Fuzzy C-Means* (*HOFCM*).

The *FCM* clustering problem is of the *NP-Hard* type, which justifies using heuristic methods for its solution. To date, improvements in the initialization phase of the *FCM* algorithm have been proposed. However, they need to be oriented to the solution of large datasets like those presented in Big Data. In this sense, the proposed algorithm reduces the number of iterations by optimizing the initial values of the membership matrix. This approach consists of three steps: a) generate a set of *n* solutions for a given dataset, applying a variant of the *K-Means* algorithm; b) select the best solution as a basis for generating the optimized membership matrix and c) solve the given dataset with *FCM*.

To validate the results of the *HOFCM* algorithm, a set of experiments composed of real and synthetic datasets were designed and executed. *HOFCM* was contrasted with the algorithms: standard *FCM*, *FCM-KMeans*, *FCM++*, and *NFCM*. Based on the experimental results, it was observed that *HOFCM* obtained a reduction in solution time in all large datasets compared to the standard *FCM* algorithm. By contrasting the results of *HOFCM* with the *FCM-KMeans*, *FCM++*, and *NFCM* algorithms, it was observed that, on average, it was faster by 1.51, 2.87, and 3.01 times, respectively.

Finally, based on the results obtained, it is possible to affirm that this research benefits users who seek to solve the problem of fuzzy clustering of large datasets such as those presented in Big Data in a reasonable time.

Contenido Pág.

Lista de Figuras

Pág.

Lista de Tablas

Pág.

Capítulo 1

Introducción

En este Capítulo se exponen las principales motivaciones que dieron origen a la presente investigación, se describe el problema de investigación, el objetivo general, los objetivos particulares, la hipótesis, así como los alcances y limitaciones de esta investigación. Finalmente, se describe el contexto que dio origen al estudio, así como la organización del documento.

1.1 Motivaciones

De acuerdo con las recientes estimaciones, se prevé que para el año 2025, el volumen de datos generados en todo el mundo supere los 180 zetabytes [1]. Este incremento exponencial de datos, se debe principalmente a dos fuentes: internet de las cosas y redes sociales [2, 3]. Tal crecimiento ha generado nuevos retos, porque actualmente no existen algoritmos eficientes que se ejecuten en tiempo polinomial para procesar estos grandes volúmenes de datos con herramientas computacionales estándar [4-6]. Algunas alternativas para resolver esta problemática, se abordan en: la Ciencia de Datos, la Minería de Datos y la Analítica de Datos. Estas disciplinas, tienen como objetivo encontrar patrones que coadyuven a la toma de decisiones basadas en los datos [7-10]. Para lograr dicho objetivo, estas ciencias se apoyan en varias técnicas, una de las principales son los algoritmos de agrupamiento o *clustering*.

El *clustering*, ha sido estudiado ampliamente en diversas áreas, tales como reconocimiento de patrones, procesamiento de imágenes, medicina, taxonomía, negocios y

minería de datos, entre otras [11-16]. En los últimos años, cobró relevancia debido al incremento exponencial de los datos en diversos ámbitos del conocimiento. Por esta razón, se convirtió en un objeto de estudio relevante para el aprendizaje automático.

1.2 Descripción del problema de investigación

La presente investigación, se ubica en el contexto de los problemas relacionados con las mejoras de los algoritmos de agrupamiento. En particular, el problema que se aborda consiste en reducir el tiempo de procesamiento o complejidad temporal del algoritmo de agrupamiento *Fuzzy C-Means* (*FCM*) propuesto por Bezdek en 1981 [17].

En la Figura 1, se describe gráficamente el problema central de esta investigación. En el eje *x*, se representa el tamaño de los *datasets* que se deben resolver; en el eje *y*, el tiempo de solución requerido. La línea verde representa la propuesta de esta investigación, la cual consiste en reducir la complejidad temporal del algoritmo *FCM*. Es importante aclarar, que la diferencia de la línea verde con respecto a la complejidad temporal de *FCM*, solamente es ilustrativa, con la intención de mostrar el concepto de la reducción de complejidad.

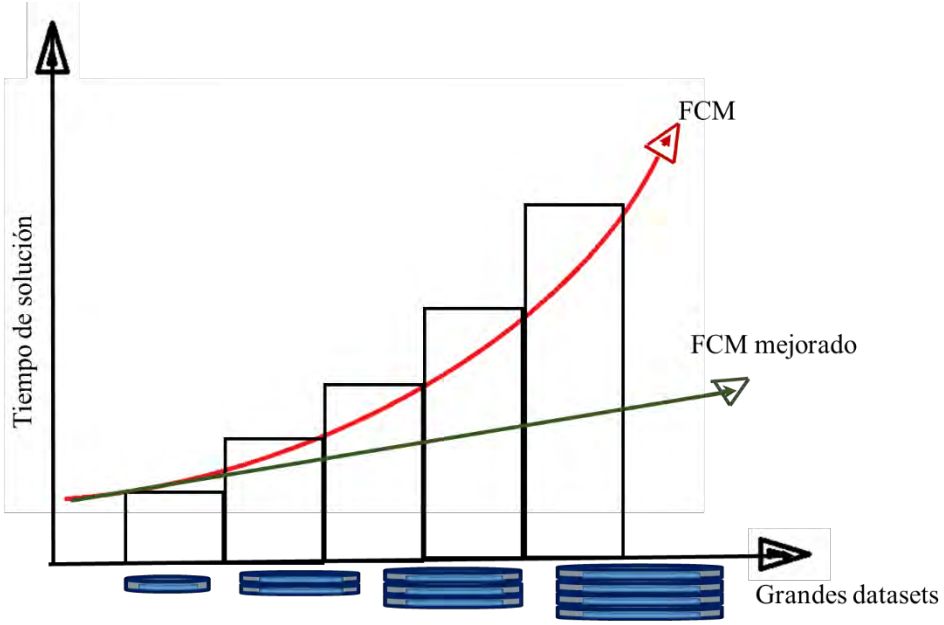

Figura 1. Complejidad temporal del algoritmo *Fuzzy C-Means* y propuesta de solución

1.3 Objetivo general

Reducir la complejidad temporal del algoritmo *Fuzzy C-Means* mediante métodos heurísticos, sin afectar significativamente la calidad de la solución.

1.3.1 Objetivos particulares

Derivados del objetivo general, se proponen los siguientes objetivos particulares:

a) Desarrollar una mejora eficiente en la fase de inicialización del algoritmo *FCM*, de manera que reduzca su complejidad temporal.

b) Implementar de manera computacional la mejora propuesta.

c) Validar los resultados de la mejora.

1.4 Hipótesis

Es factible desarrollar una mejora eficiente en la fase de inicialización del algoritmo *Fuzzy C-Means* para la solución de grandes *datasets* mediante métodos heurísticos, sin afectar significativamente la calidad de la solución.

1.5 Alcances

Se consideran los siguientes alcances:

a) Una variante de *FCM* cuya complejidad temporal es menor a la de *FCM* estándar.
b) La mejora propuesta para este algoritmo, supera a otras mejoras expuestas en publicaciones anteriores en la fase de inicialización.

1.6 Limitaciones

A continuación, se enlistan las limitaciones de la presente investigación:

a) La investigación está enfocada en el algoritmo *Fuzzy C-Means*.
b) La implementación de la variante del algoritmo es probada con el equipo disponible en el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET).

c) Los experimentos, para validar el enfoque de solución, son realizados con *datasets* reales de repositorios reconocidos por la comunidad científica, por ejemplo UCI [18] y se generaron *datasets* de prueba sintéticos exprofeso.

1.7 Contexto de la investigación

El contexto de esta investigación se enmarca dentro del problema de los algoritmos de agrupamiento. Tradicionalmente, los algoritmos de agrupamiento se dividen en jerárquicos y particionales [19]. A su vez, los algoritmos de agrupamiento particionales, se dividen en dos: duros y suaves [20-22]; del primero, su principal exponente es el algoritmo *K-Means* [23]; y del segundo el algoritmo *FCM* [17].

Existen mejoras realizadas en diferentes etapas del algoritmo de agrupamiento *K-Means*, desarrolladas con éxito en el CENIDET. A continuación, se mencionan las investigaciones más recientes:

1) Tesis de doctorado

   Almanza Ortega Nelva Nelly, (2018). "Desarrollo de heurísticas para la mejora del algoritmo K-Means en las fases de clasificación y convergencia".

   En esta investigación, se presentan dos mejoras al algoritmo *K-Means*. La primera, denominada *A-Means*, mejora la fase de clasificación mediante la asignación temprana de objetos a grupos, evitando subsecuentes cálculos de distancia objeto-centroide. La segunda, denominada *O-K-Means*, mejora la fase de convergencia al detener el algoritmo, cuando el total de objetos que cambian de grupo en una iteración es menor a un umbral.

2) Artículo

   Pérez Ortega Joaquín, Almanza Ortega Nelva Nelly, Romero David, (2018). "Balancing effort and benefit of K-Means clustering algorithms in Big Data realms".

   En este artículo se propone un criterio para equilibrar el tiempo de procesamiento y la calidad de la solución del algoritmo *K-Me*ans, cuando se aplican a conjuntos de

datos del tipo *Big Data*. Este artículo se deriva de la tesis de doctorado, mencionada previamente.

1.8 Organización del documento

El presente documento posee la siguiente estructura: en el Capítulo 2, se exponen las principales investigaciones que tienen relación con esta tesis doctoral; en el Capítulo 3, se presenta la propuesta de mejora del algoritmo *FCM*; en el Capítulo 4, por una parte, se describe la implementación del algoritmo propuesto y por otra parte se muestra la validación experimental de los resultados; en el Capítulo 5, se presenta el análisis de los resultados y las conclusiones derivadas de la presente investigación, así como algunas propuestas para continuar con nuevas investigaciones respecto del algoritmo *FCM*.

Capítulo 2

Algoritmo *Fuzzy C-Means*

En el presente Capítulo se exponen los antecedentes del algoritmo *FCM*. Asimismo, se describe de manera formal el algoritmo y se reportan en forma sintetizada las publicaciones recientes y las de mayor impacto en la comunidad científica relacionadas con las mejoras más relevantes del algoritmo.

2.1 El problema de agrupamiento

En la teoría tradicional de conjuntos, un objeto puede pertenecer o no a un conjunto dado. Sin embargo, existen problemas en los cuales la pertenencia de un objeto a un conjunto no se puede establecer de manera precisa. Una manera de modelar dicha pertenencia, es a través de los algoritmos de agrupamiento. Tradicionalmente, estos algoritmos se dividen en jerárquicos y particionales [19]. A su vez, los algoritmos de agrupamiento particionales, se dividen en duros y difusos.

En esta investigación, se asume que los atributos de los objetos de un conjunto son medibles. Por lo tanto, a veces se hará referencia a los objetos como puntos, de acuerdo con el contexto.

2.1.1 Agrupamiento particional duro

De los algoritmos de agrupamiento particionales duros, uno de los más relevantes, ampliamente estudiado y utilizado, es el algoritmo *K-Means* [24-26]. Este algoritmo, es un método iterativo que consiste en particionar un conjunto de n objetos en $k \geq 2$ grupos, de tal manera que los objetos

de un grupo sean similares entre si y diferentes a los de otros grupos [27]. La formulación del problema *K-Means* como un problema de optimización, se describe a continuación:

Sea $X = \{x_1, \ldots, x_n\}$ el conjunto de n objetos a particionar por un criterio de similaridad, donde $x_i \in \mathfrak{R}^d$ para $i = 1, \ldots, n$ y $d \geq 1$ es el número de dimensiones. Además, sea $k \geq 2$ un entero y $K = \{1, \ldots, k\}$. Para una k-partición $P = \{G(1), \ldots, G(k)\}$ de X, se denota que v_j es el centro del grupo $G(j)$, para $j \in K$ y sea $V = \{v_1, \ldots, v_k\}$ y $U = \{u_{11}, \ldots, u_{ij}\}$.

En la Expresión 1, se plantea el problema de agrupamiento como un problema de optimización [28].

$$P: \text{minimizar } z(U,V) = \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij} D(x_i, v_j) \tag{1}$$

$$\text{Sujeto a } \sum_{j=1}^{k} u_{ij} = 1, \quad \text{para } i = 1,...,n,$$

$$u_{ij} = 0 \text{ ó } 1, \text{ para } i = 1, \ldots, n \text{ y } j = 1, \ldots, k,$$

Donde $u_{ij} = 1 \Leftrightarrow$ el objeto x_i pertenece al grupo $G(j)$ y $D(x_i, v_j)$ denota la distancia Euclidiana entre x_i y v_j para $i = 1, \ldots, n$ y $j = 1, \ldots, k$. El pseudocódigo del algoritmo *K-Means* estándar se muestra en el Algoritmo 1.

Algoritmo 1: *K-Means* estándar

Entrada: X, V, k,
Salida: V, U

Inicialización:
  X: = $\{x_1, \ldots, x_n\}$;
  V: = $\{v_1, \ldots, v_k\}$;
Clasificación:
  Para $x_i \in X$ and $v_k \in V$ {
    Calcular la distancia Euclidiana de cada x_i a los k centroides;
    Asignar el objeto x_i al centroide v_k más cercano; }
Cálculo de centroides:
  Calcular los centroides v_k;
Convergencia:
  Si V: = $\{v_1, \ldots, v_k\}$ no cambia en dos iteraciones consecutivas:
    Detener el algoritmo;

13 En caso contrario:
14 Ir a **Clasificación**
15 **Fin del algoritmo**

2.1.2 Agrupamiento particional difuso

La teoría de conjuntos difusos propuesta por Zadeh [29] en 1965, aportó una idea de incertidumbre de pertenencia que fue descrita mediante una función de pertenencia. La teoría del análisis de grupos se propuso en la investigación de Bellman, Kalaba y Zadeh [30] y en Ruspini [31, 32] se acuñó el concepto de partición difusa, más específicamente de algoritmo de agrupamiento difuso. Estos autores, dieron la pauta a posteriores investigaciones acerca de agrupamiento difuso. En 1973, Dunn [33] extendió el significado de agrupamiento duro a conceptos preliminares de *fuzzy Means*. Por otra parte Bezdek [17] , generalizó el enfoque de Dunn para obtener una familia infinita de algoritmos conocida como algoritmos *c-means* difusos. La idea básica es que sea $X = \{x_1, \ldots, x_n\}$ el conjunto de n objetos a particionar por un criterio de similitud, donde $x_i \in \Re^d$ para $i = 1, \ldots, n$ y c es el número de grupos donde $2 \leq c < n$.

El problema de agrupamiento difuso se formuló como un problema de optimización [31] minimizando una función, como se muestra en la Expresión 2.

$$P: \text{minimizar } J_m(U,V) = \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}{}^{m} D(x_i, v_j) \tag{2}$$

Donde $U = \mu_{ij}$ es la matriz que contiene los valores de los grados de pertenencia de cada objeto i a cada grupo j; $V = \{v_1, \ldots, v_c\}$ es el conjunto de centroides, donde v_j es el centroide del grupo j; m es el exponente de ponderación que altera los grados de pertenencia de cada objeto, $m > 1$ y $D(x_i, v_j)$, indica la distancia Euclidiana entre el objeto x_i y el centroide v_j para i=1, …, n y j=1, …, c.

La minimización de la función objetivo J_m se realiza mediante la actualización iterativa de la matriz de pertenencia utilizando las Expresiones 3 y 4.

$$v_j = \frac{\sum_{i=1}^{n} (u_{ij})^m x_i}{\sum_{i=1}^{n} (u_{ij})^m}; \quad 1 \le j \le c \tag{3}$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\| x_i - v_j \|^2}{\| x_i - v_k \|^2} \right)^{\frac{1}{m-1}}} \qquad 1 \le i \le n; \quad 1 \le j \le c \tag{4}$$

Donde x_i y v_j son vectores que pertenecen a un espacio $\Re^d$, tal como se representan en la Expresiones 5 y 6.

$$x_i = (x_1, x_2, \ldots, x_d), \quad 1 \le i \le n \tag{5}$$

$$v_j = (v_1, v_2, \ldots, v_d), \quad 1 \le j \le c \tag{6}$$

Las restricciones del agrupamiento difuso se formalizan en las Expresiones 7-9:

$$u_{ij} \in [0,1], \qquad 1 \le j \le c, \qquad 1 \le i \le n \tag{7}$$

$$\sum_{j=1}^{c} u_{ij} = 1, \qquad 1 \le i \le n \tag{8}$$

$$0 < \sum_{i=1}^{n} u_{ij} < n, \qquad 1 \le j \le c \tag{9}$$

La Expresión 7, indica que el grado de pertenencia de un objeto *i* a un grupo *j* debe estar entre 0 y 1; la Expresión 8 define que la suma de los grados de pertenencia de un objeto *i* a los distintos grupos ha de ser igual a 1; y la Expresión 9 indica que la suma de todos los grados de pertenencia en un grupo tiene que ser mayor a 0 y menor que *n*, es decir, no debe haber grupos vacíos y un solo grupo.

FCM está formado por cuatro fases, como se muestra en la Figura 2 y también en el Algoritmo 2, se muestra el pseudocódigo.

Figura 2. Fases del algoritmo *Fuzzy C-Means*

1. **Inicialización**. Para un conjunto de objetos a ser particionado, se necesita definir el número de grupos, el valor del exponente de ponderación, determinar el valor de un umbral (ε) para terminar las iteraciones o determinar el número máximo de iteraciones y generar la matriz que contiene los grados de pertenencia de cada objeto a cada grupo aleatoriamente. Se destaca que los valores del exponente de ponderación se encuentran en el rango de 1.25 a 2 [34-36]. Sin embargo, este valor puede ser mayor a 2, inclusive hasta 10 [37]. Asimismo, el valor del umbral para terminar las iteraciones, puede tener diferentes rangos, por ejemplo 0.01, 0.03, 0.00001, 0.000001 [22, 38-40].

2. **Cálculo de centroides**. Calcular el centroide de cada grupo (Ver Expresión 3).

3. **Clasificación**. Se calcula el grado de pertenencia de cada objeto a cada uno de los centroides de los grupos creados (Ver Expresión 4).

4. **Convergencia**. Existen diferentes criterios de paro, entre los cuales, los más usados son los siguientes: a) parar el algoritmo cuando se alcanza un número máximo de iteraciones; b) parar el algoritmo cuando el valor absoluto de la diferencia máxima de los grados de pertenencia en dos iteraciones consecutivas sea menor que un umbral dado; c) parar el algoritmo cuando el valor absoluto de la diferencia máxima entre centroides en dos iteraciones consecutivas sea menor que un umbral dado. Mientras que el criterio de parada sea falso se repiten los pasos dos y tres del algoritmo.

Algoritmo 2: *FCM* estándar

Entrada: X, U, c, m, ε
Salida: V, U

Inicialización:
  t: = 0;
  $U^{(t)}$: = $\{\mu_{11}, \ldots, \mu_{ij}\}$; se genera aleatoriamente
Cálculo de centroides:
  Calcular los centroides con Ex. (3);
Clasificación:
  Actualizar y calcular la matriz de pertenencia con Ex. (4);
Convergencia:
  Si max [abs($\mu_{ij}^{(t)} - \mu_{ij}^{(t+1)}$)] < ε:
    Detener el algoritmo;
  En caso contrario:
    $U^{(t)}$: = $U^{(t+1)}$ and t: = t + 1;
  Ir a **Cálculo de centroides**
Fin del algoritmo

A partir de los estudios pioneros realizados por Ruspini [31, 32], Dunn [33] y Bezdek [17], numerosas investigaciones se dedicaron a encontrar una c-partición difusa de $\boldsymbol{X}$ que resuelva $\boldsymbol{P}$ definida en la Expresión 2.

En [41-43] se afirmó que la complejidad temporal del algoritmo *FCM* es $O(ndc^2t)$, donde n es el número de objetos, d el número de dimensiones, c el número de grupos y t el número de iteraciones.

El agrupamiento es uno de los problemas *NP-hard* más estudiados en la comunidad científica [44]. No existen algoritmos eficientes que se ejecuten en tiempo polinomial con respecto a grandes *datasets* para resolverlo. Estos problemas se consideran computacionalmente *intratables*. Por lo tanto, mejorar el algoritmo de agrupamiento de *FCM* es un problema de investigación abierto y relevante.

2.2 Trabajos relacionados

En esta sección, de describen las publicaciones relacionadas con el algoritmo *FCM* antes y posterior a Bezdek. Adicionalmente, se describen mejoras del algoritmo en relación con las fases del algoritmo *FCM* estándar.

2.2.1 Trabajos relacionados antes de Bezdek

Los métodos de agrupamiento convencionales, restringen la posibilidad de que cada objeto del conjunto pertenezca exactamente a un grupo. La teoría de conjuntos difusos, propuesta por Zadeh en 1965 [29] relaja la restricción de que el grado de pertenencia de un objeto a un conjunto valga 0 ó 1. A continuación, se explica esta teoría: Sea X un espacio de puntos (objetos) con un elemento genérico de X denotado por x. Así, $X = \{x\}$. Un conjunto difuso (clase) A en X se caracteriza por una función de pertenencia $f_A(x)$ que asocia con cada objeto en X un número real en el intervalo [0, 1], con el valor de $f_A(x)$ en x que representa el grado de pertenencia de x en A. Por lo tanto, cuanto más cercano sea el valor de $f_A(x)$ a la unidad, mayor será el grado de pertenencia de x en A.

En 1966, Bellman, Kalaba y Zadeh [30] propusieron la teoría de análisis de grupos, la cual consiste en: Dado dos conjuntos disjuntos A y B en un espacio de puntos. Se tiene n puntos $\alpha_1, \ldots, \alpha_n$ que pertenecen a A y m puntos $\beta_1, \ldots, \beta_m$ que pertenecen a B. El problema es construir estimaciones de las funciones de pertenencia de A y B con base en el conocimiento de las muestras $\alpha_1, \ldots, \alpha_n$ de A y $\beta_1, \ldots, \beta_m$ de B. Se puede intentar estimar f_A sin hacer uso de β_j, $j = 1, \ldots, m$. Sin embargo, en general, una estimación de este tipo no sería tan buena como la que se emplea tanto en α's y β's. Esto es consecuencia de una dependencia implícita o explícita entre A y B a través de la cual el conocimiento de β's aporta cierta información sobre f_A. Lo mismo se aplica a las estimaciones de f_B. La regla heurística sugerida es simplemente una forma de construir estimaciones de f_A y f_B, dando $\alpha_1, \ldots, \alpha_n$ y $\beta_1, \ldots, \beta_m$,.

En 1969, Ruspini [31] introdujo el término de agrupación difusa. Sin embargo, donde desarrolla el modelo matemático de este tipo de agrupamiento, fue en su estudio publicado en

1970 [32]. En éste se formula por primera vez, el agrupamiento difuso como un problema de optimización, basado en una función objetivo $J_R(U)$, tal como se presenta en la Expresión 10.

$$P: \text{minimizar } J_R(U) = \sum_{k=1}^{n} \sum_{j=1}^{n} (\alpha \sum_{i=1}^{c} (u_{ij} - u_{ik})^2 - D^2_{ij})^2 \quad (10)$$

Donde u_{ik} es grado de pertenencia del k-ésimo objeto al i-ésimo agrupamiento y D_{ij} es la distancia del objeto i al objeto j. σ es una constante real, $2 < c < n$ determinada a priori. La función objetivo como medida de calidad del agrupamiento está basada en la densidad local. Se destaca que, aunque la función objetivo es complicada y difícil de interpretar, lo importante de la investigación de Ruspini, es la introducción de la idea de la c-partición difusa a problemas de agrupamiento.

En 1973, Dunn [33] definió la primera generalización difusa del problema convencional de partición de mínima varianza y derivó las condiciones necesarias para minimizar la función objetivo J_D definida en la Expresión 11. Usó estas condiciones para desarrollar un esquema de iteración de *Picard* para la optimización iterativa de J_D y lo llamó *Fuzzy ISODATA*, en deferencia a su relación con el proceso *Hard ISODATA* de Ball y Hall [45].

$$P: \text{minimizar } J_D(U,V) = \sum_{i=1}^{n} \sum_{j=1}^{k} u_{ij}^2 D(x_i, v_j) \quad (11)$$

Nótese que, a diferencia de la propuesta de Bezdek [17], el exponente de ponderación que altera los grados de pertenencia de cada objeto es fijo.

2.2.2 Trabajos de Bezdek

En esta sección, se reportan en forma sintetizada las publicaciones realizadas por Bezdek, desde su tesis doctoral en 1973, hasta 1980. Estas publicaciones, son los antecedentes a la publicación

de 1981 [17], misma que es objeto de estudio de esta investigación. Dichas publicaciones se presentan a continuación de manera cronológica:

- En su tesis doctoral [46], sentó las bases matemáticas para el agrupamiento difuso planteado como un problema de optimización. En esta tesis, considera como base las investigaciones de Dunn[33].
- En los artículos publicados en 1974 y 1975 [47-49], extrajo de su tesis doctoral un ejemplo con el conjunto de datos *IRIS*, con el cual resuelve el agrupamiento con el algoritmo *Hard ISODATA* y *Fuzzy ISODATA*. Se destaca el índice de validación Coeficiente de Partición.
- En 1976 [50], se publicó la aplicabilidad de los algoritmos de agrupamiento *Fuzzy ISODATA* para la reducción de la dimensionalidad de los conjuntos de datos con valores binarios y el diagnóstico médico computarizado.
- En su artículo publicado en 1978 [51], se exploraron algunas conexiones entre particiones difusas y relaciones de similitud. Se propuso una nueva definición de transitividad para relaciones difusas. Esta noción de transitividad, también vincula la desigualdad triangular con descomposiciones convexas de relaciones de similitud difusa de una manera que puede generar nuevas técnicas para la agrupación difusa.
- En su artículo de 1979 [52], se investigaron algunas propiedades algebraicas y geométricas de los espacios de partición difusos. En particular, se obtuvieron sus dimensiones y se describieron una serie de algoritmos, para efectuar descomposiciones convexas.
- En 1980 [53], estableció la convergencia del algoritmo *Fuzzy ISODATA*. Se aplicó la teoría de Zangwill [54], para demostrar que las secuencias arbitrarias generadas por la iteración de *Picard*, siempre terminan en un mínimo local, o en el peor de los casos, siempre contiene una subsecuencia que converge a un mínimo local de la función objetivo de mínimos cuadrados.

Otras publicaciones de Bezdek después de 1981 se pueden encontrar en los siguientes artículos [22, 35, 55-61].

2.2.3 Trabajos relacionados con las distintas fases del algoritmo *FCM*

Las publicaciones que a continuación se presentarán, fueron seleccionadas por su relevancia en cuanto al presente tema de investigación. Con este fin, se consideró importante conocer y entender las mejoras que ha reportado la comunidad científica en cada una de las fases del algoritmo.

Fase de Inicialización

En [62], se obtuvo ventaja de la similitud entre el algoritmo *K-Means* y el *FCM*. En particular, para la selección de centroides iniciales, se adaptó una mejora al algoritmo *K-Means*, nombrada *K-Means*++ [63]. El algoritmo propuesto se nombró *FCM*++. En esta propuesta, se adicionó al algoritmo *K-Means* ++, el parámetro *p* al cálculo de la distancia (D^p). Dicho parámetro, permite controlar el factor de propagación del algoritmo. Posteriormente, a esta selección de centroides, se continúa con el algoritmo *FCM* estándar. Sin embargo, esta mejora presenta algunas limitaciones: la primera, es para los *datasets IRIS* y *WINE*, donde el número de iteraciones es superior respecto al *FCM* estándar para c=8 y c=10 respectivamente; y la segunda, es con respecto a la elección del parámetro *p*, ya que su eficiencia depende de la experiencia del usuario. Es importante destacar, que este factor no se encuentra definido en el algoritmo *K-Means*++. La descripción del algoritmo *K-Means*++ se proporcionará en la Sección 3.3.1.

En [64], se utilizó el algoritmo *K-Means* para generar los centroides finales, los cuales, a través de un proceso de transformación denominado *One-Hot encoding*, generan la matriz de pertenencia que requiere el algoritmo *FCM*. Para llegar a dicha codificación, esta investigación propuso seleccionar los atributos por medio de ponderación de entropía y transformación *Box-Cox*. De tal manera que, de 2,170 dimensiones, únicamente se utilizaron 10. En la parte experimental, emplearon un *dataset* denominado *CSI 800* que almacenó el histórico de precios diarios de acciones. En los resultados obtenidos se observó una disminución del tiempo de ejecución del 24% con respecto al *FCM* estándar.

En [37], se retomó la estrategia del algoritmo *FCM*++ [62]. La diferencia con el *FCM*++, radica en la consideración de la información que provee la matriz de pertenencia para

la selección de los siguientes centroides. En la parte experimental, se mostró la competitividad del algoritmo propuesto denominado *NFCM*, con respecto al *FCM++*. Para ello utilizaron dos *datasets*: *SPAM* e *IRIS*. En *IRIS*, se observó que el algoritmo *NFCM* presenta cierta ventaja en el número de iteraciones y el tiempo de CPU para cuando $c>4$. Mientras que el *dataset SPAM*, cuando $c=8$ y $c=9$ el algoritmo *FCM++* tiene mayor competitividad.

Las limitaciones que presentan las investigaciones reportadas en [37, 62, 64] son: la falta de descripción explícita de la variante con respecto al algoritmo *FCM* estándar; y la inexistencia de comparaciones entre las propuestas con dicho algoritmo.

La Figura 3, muestra el enfoque de solución de las tres investigaciones descritas en la fase de inicialización del algoritmo *FCM* encontradas hasta la fecha en la literatura.

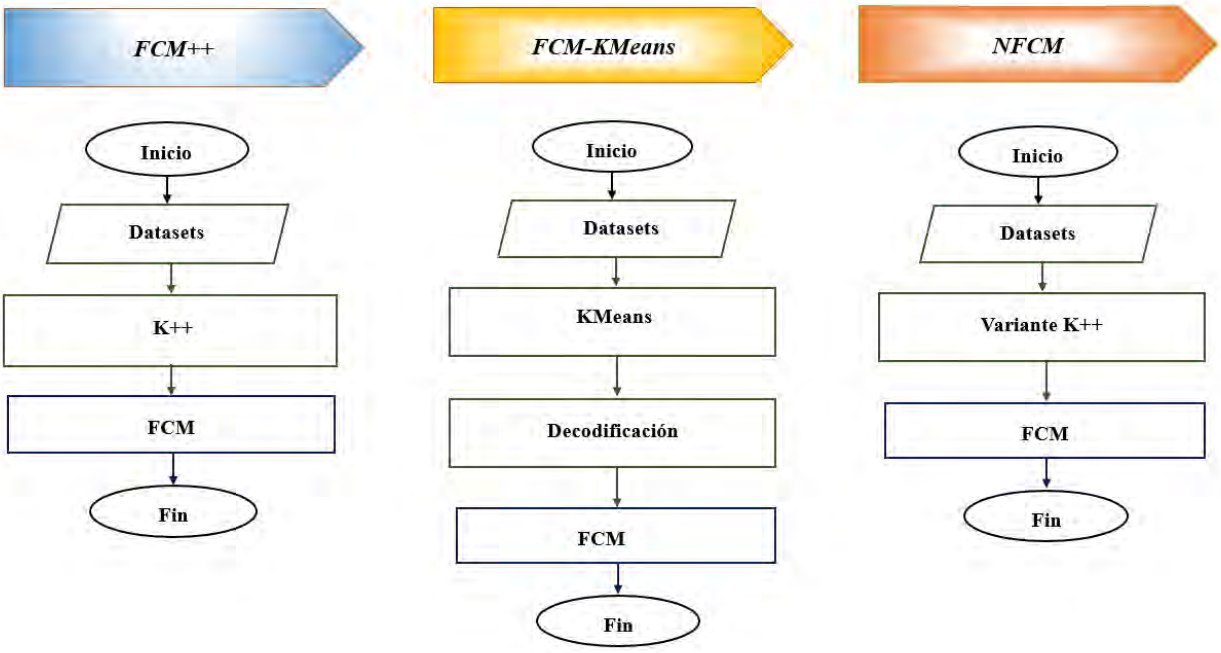

Figura 3. Propuestas de solución en la fase de inicialización de la literatura

Como se puede observar, se han propuesto pocas mejoras en la etapa de inicialización del algoritmo *FCM*, las cuales, no están orientadas a la solución de grandes *datasets* como las que

se presentan en el *Big Data*. En contraste, el algoritmo que se propone en esta investigación, tiene la finalidad de reducir el número de iteraciones, al optimizar la matriz de pertenencia inicial a partir de la mejor solución de una muestra de *n* soluciones de un algoritmo híbrido denominado *O-K-Means++* enfocado a la solución de grandes *datasets*.

Otras fases del algoritmo

En este apartado, se describirán mejoras que se ubican en otras fases del algoritmo. Es importante mencionar, que las principales investigaciones reportadas en la presente investigación, están enfocadas a mejorar la fase de clasificación del algoritmo.

En [65], se propuso reducir el tiempo de complejidad temporal del algoritmo *FCM*. El algoritmo original, alterna entre el cálculo de los centroides y el cálculo de grados de pertenencia de los objetos hacia los centroides. El tamaño de las estructuras de datos para el cálculo antes mencionado, está en el orden del conjunto de datos original, un tamaño prohibitivo si esta técnica se aplica a *datasets* grandes. La implementación propuesta en este artículo, es eliminar el almacenamiento de esta estructura de datos, al combinar dos actualizaciones en una única actualización de los centroides. Este cambio afecta significativamente el tiempo de ejecución asintótico ya que el nuevo algoritmo es lineal con respecto al número de grupos, mientras que el original es cuadrático. La complejidad reportada en esta publicación es de $O(ncd)$ en una iteración. Esta implementación realizada en segmentación de imagen requiere solo 0.5 s/iteración mientras que el sistema de referencia original requirió 2.1 segundos.

En [66], se presentaron resultados del algoritmo denominado *BCFCM* (*Bías Corrected Fuzzy C-Means*) enfocado hacia la segmentación adaptativa y corrección de intensidad de imágenes de resonancias magnéticas. La principal aportación de este algoritmo es la modificación de la función objetivo del algoritmo *FCM* estándar. La función objetivo de *FCM* estándar, se minimiza cuando se asignan valores de pertenencia altos a pixeles cuyas intensidades están cerca del centroide de su grupo y se asignas valores de pertenencia bajos cuando los datos de pixeles están lejos del centroide. Para realizar la modificación de la función objetivo, se introdujo un parámetro α que permitió que el etiquetado de un pixel sea influenciado por las

etiquetas de su vecindad inmediata. El efecto de vecindad, actuó como un regularizador y sesgó la solución hacia un etiquetado por partes homogéneas. Dicho parámetro, se normalizó por la técnica de *signal to noise ratio* (SNR). El SNR se define como la relación de potencia de la señal con la potencia de ruido, a menudo expresada en decibelios.

En [67], se mostró que una asignación de ponderación a los atributos, puede mejorar el rendimiento de la agrupación *FCM*. La asignación de peso se dio con la técnica de descenso del gradiente. Los experimentos con seis *datasets* de UCI demostraron la mejora del rendimiento del agrupamiento difuso. La complejidad temporal del algoritmo propuesto, denominado *WFCM*, es $O(cn^2)$, donde n es el número de objetos y c es una constante asociada con el número de dimensiones. La complejidad del tiempo dependió de las iteraciones para la convergencia. Esta técnica es convergente si los pasos son apropiadamente pequeños. Pero los pasos demasiado pequeños, harán que la tasa de convergencia sea muy lenta.

En [68], al igual que en [66], se modificó la función objetivo considerando un parámetro denominado S como medida de similitud local, que garantizó tanto la inmunidad al ruido como la preservación de detalles para la imagen de acuerdo a sus autores. La reducción de la complejidad reportada en esta investigación es $O(qcI)$, donde c es el número de grupos, q el tiempo de segmentación e I el número de iteraciones. El algoritmo propuesto, denominado *FGFCM* (*Fast Generalized Fuzzy C-Means*) reportó menor cantidad de iteraciones que se reflejó en menor tiempo de ejecución. Por ejemplo, para una imagen determinada, la cantidad de iteraciones se redujo de 277 en el *FCM* estándar a 223 en *FGFCM*, lo que representó 1.76% menor en el número de iteraciones. Para esta misma imagen, el tiempo de ejecución se redujo de 37.58 segundos a 0.2407 segundos, lo cual representó una reducción en tiempo de ejecución del 99.36%.

En [34], se propusieron dos ponderaciones, una para los atributos y otra para los grupos. La ponderación de atributos se realizó de forma local, de manera que los atributos tuvieran diferentes pesos dependiendo de su importancia en los grupos; el peso de los grupos se calculó dinámicamente considerando la importancia de las características de cada grupo. Además, el

algoritmo propuesto utilizó una función objetivo basada en una métrica de distancia no Euclidiana. La ventaja de utilizar esta métrica de distancia es que el ruido y los valores atípicos no pueden afectar mucho el proceso de ponderación de características. Los resultados de los experimentos en *datasets* del mundo real, así como *datasets* sintéticos confirmaron que el algoritmo propuesto tiene un rendimiento muy prometedor, y no es sensible a la inicialización de los centros de grupos, incluso para una mala inicialización, y demostró una mejor agrupación que los competidores. La complejidad computacional reportada es de $4tNMK$ donde t es el número de iteraciones, N el número de objetos, M el número de dimensiones y K el número de grupos. La investigación, reportó diferentes resultados considerando las métricas propuestas para los diferentes *datasets*. Sin embargo, no reportó resultados respecto el tiempo de ejecución y tampoco la cantidad de iteraciones.

Estas mejoras al algoritmo *FCM* aquí descritas, están enfocadas principalmente a un dominio específico. A diferencia de la propuesta que se pretende desarrollar con la presente investigación, la cual tiene un enfoque de dominio general. Se destaca que los *datasets* utilizados en la presente investigación, son de mayor tamaño que los reportados en las mejoras de esta sección.

Finalmente, se sabe que existen otro tipo de mejoras del algoritmo *FCM*, por ejemplo, las que tienen un enfoque paralelo. Algunos de los estudios, utilizan diferentes plataformas y arquitecturas, tales como: interfaz de paso de mensaje (MPI) [69], basadas en la nube [70, 71], GPU [72, 73], Spark [74, 75], MAP Reduce [6, 76, 77], OpenCI [78] y OpenMPl [79-81].

Capítulo 3

Propuesta de mejora del algoritmo *FCM*

De acuerdo con la literatura especializada, se ha demostrado que el algoritmo de agrupamiento *FCM* es *NP-hard*. Por esta razón, se justifica el uso de heurísticas como alternativa de solución. En este Capítulo, se expone el funcionamiento del nuevo algoritmo denominado *Hybrid OK-Means Fuzzy C-Means* (*HOFCM*), el cual se desarrolló e implementó con el propósito de mejorar la eficiencia del algoritmo *FCM* en su fase de inicialización.

3.1 Enfoque general de solución

En esta sección, se presenta el enfoque general con el cual se desarrolló el nuevo algoritmo que permite reducir la complejidad temporal del algoritmo *FCM*. La Figura 4, muestra las etapas del proceso que componen el enfoque de solución. Dicho proceso consta de cinco etapas. En la primera etapa, se estudió y analizó el comportamiento del algoritmo *FCM*; en la segunda etapa, se obtuvieron las observaciones del análisis de las principales mejoras del algoritmo; en la tercera etapa, se propuso la mejora del algoritmo; en la cuarta etapa, se implementó el nuevo algoritmo y se validaron los resultados; en la quinta etapa, se muestra el análisis de los resultados obtenidos de esta investigación.

En el presente capítulo, se exponen la primera, segunda y tercera etapas, correspondientes al estudio y análisis del comportamiento del algoritmo FCM, las observaciones del análisis de las principales mejoras del algoritmo y la propuesta de mejora del algoritmo *FCM*. El resto de las etapas, se expondrán en los siguientes capítulos.

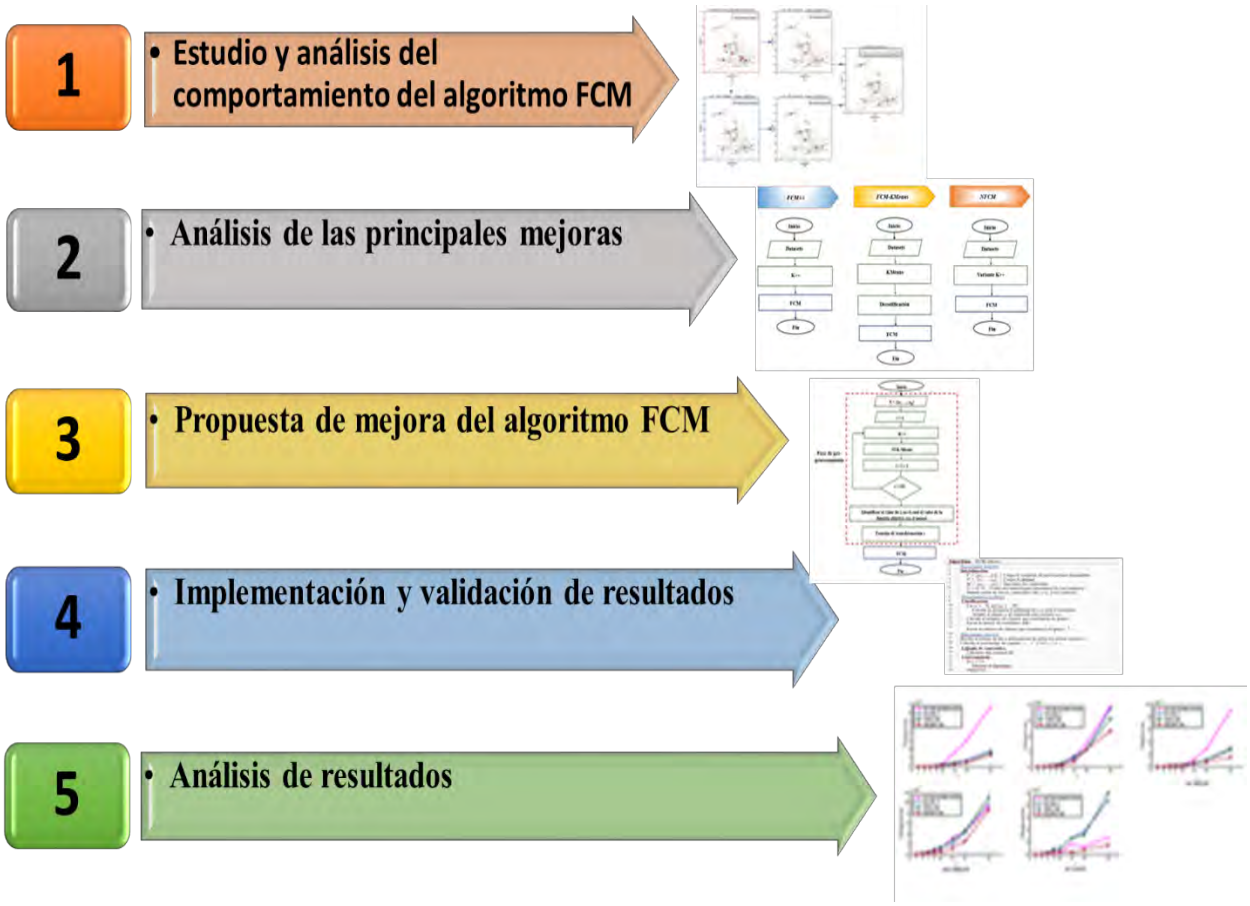

Figura 4. Enfoque general de solución

3.2 Estudio y análisis del comportamiento del algoritmo *FCM*

Esta etapa, permitió identificar el modelo matemático con el cual se da solución al algoritmo de Bezdek, las fases del algoritmo, los parámetros de entrada y salida. Lo anterior, ya se describió en el Capítulo 2. El comportamiento dinámico del algoritmo FCM, se muestra a continuación.

3.2.1 Comportamiento dinámico del algoritmo *FCM*

El algoritmo *FCM* es sensible a los valores iniciales de la matriz de pertenencia. Encontrar una buena configuración inicial es de gran importancia, porque posibilita que el algoritmo termine en menos iteraciones. En la Figura 5, se muestra el Ejemplo 1, en el cual se observan dos alternativas para seleccionar los centroides iniciales para ejecutar el algoritmo *FCM*. La primera, es generar los centroides de manera aleatoria, panel (a) y (b); y la segunda ejecutar el algoritmo *K-Means*, panel (c) y (d), cuyos centroides finales pasan como centroides iniciales al *FCM*. En el panel (e), se presenta el contraste de soluciones del algoritmo *FCM*, cuyos centroides iniciales son aleatorios y generados por el *K-Means*. El *dataset* utilizado fue *URBAN,* el cual contiene las coordenadas de longitud y latitud de accidentes de tráfico en las zonas urbanas de Gran Bretaña [18], sus valores son: n = 360,177, d = 2. Para el ejemplo se considera un valor de c = 8.

De la Figura 5, se desprenden las siguientes observaciones:

Observación 1. La posición de los centroides finales para los algoritmos *K-Means* y *FCM* estándar, cuando se ejecutan con centroides iniciales generados de manera aleatoria y centroides generados del *K-Means* son, por lo general, aproximadamente los mismos, como se muestra en el panel (e).

Observación 2. El tiempo de solución del *FCM*, para la primera alternativa, se tiene con un valor de 60,506.91 ms, panel (b) y en la segunda alternativa, con un tiempo de 26,120.84 ms, panel (d). Este último valor es el resultado de sumar los tiempos de ejecución del *K-Means* y *FCM* de los paneles (c) y (d). Con ello se tiene una reducción de tiempo del 56.83%.

Observación 3. La diferencia en el valor de la función objetivo es poco significativa, comparado con la ganancia en la solución del algoritmo.

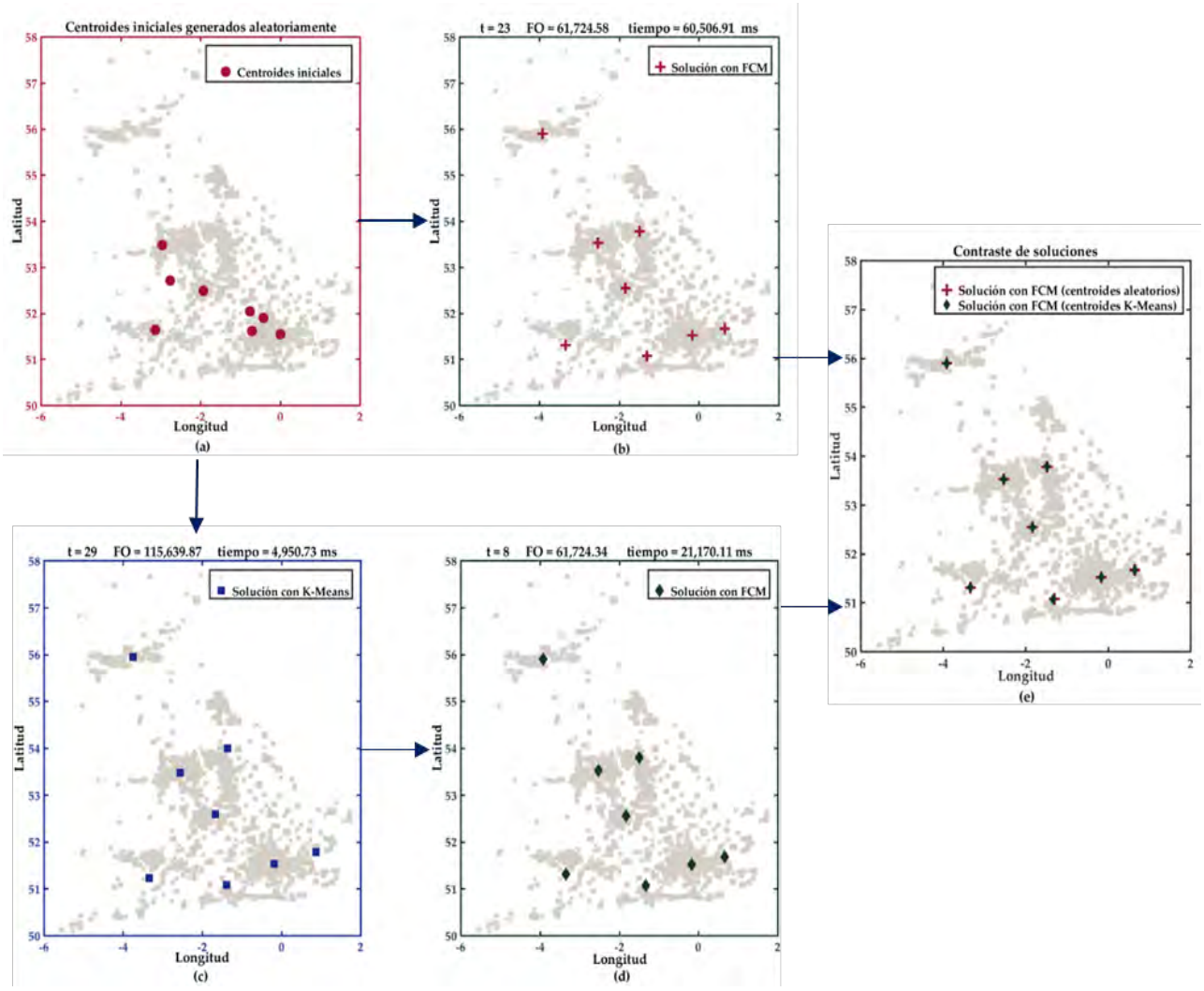

Figura 5. Centroides iniciales y finales de los algoritmos *K-Means* y *FCM* estándar con el dataset *URBAN* del Ejemplo 1

Bajo la misma idea de la Figura 5, en la Figura 6, se presenta el Ejemplo 2, del comportamiento dinámico con los algoritmos *K-Means* y *FCM* con diferentes centroides iniciales aleatorios.

De la Figura 6, se desprende la siguiente observación:

Observación 4: Los centroides finales del algoritmo *K-Means*, que pasan como parámetro de entrada al *FCM*, no siempre favorecen a la disminución de la complejidad temporal del *FCM*. En el panel (d), se observa que los tiempos de ejecución, considerando también los tiempos de ejecución del algoritmo *K-Means*, se incrementaron en 41% comparados con los del panel (b). Sin embargo, la calidad de la solución fue mejor en 33.5%.

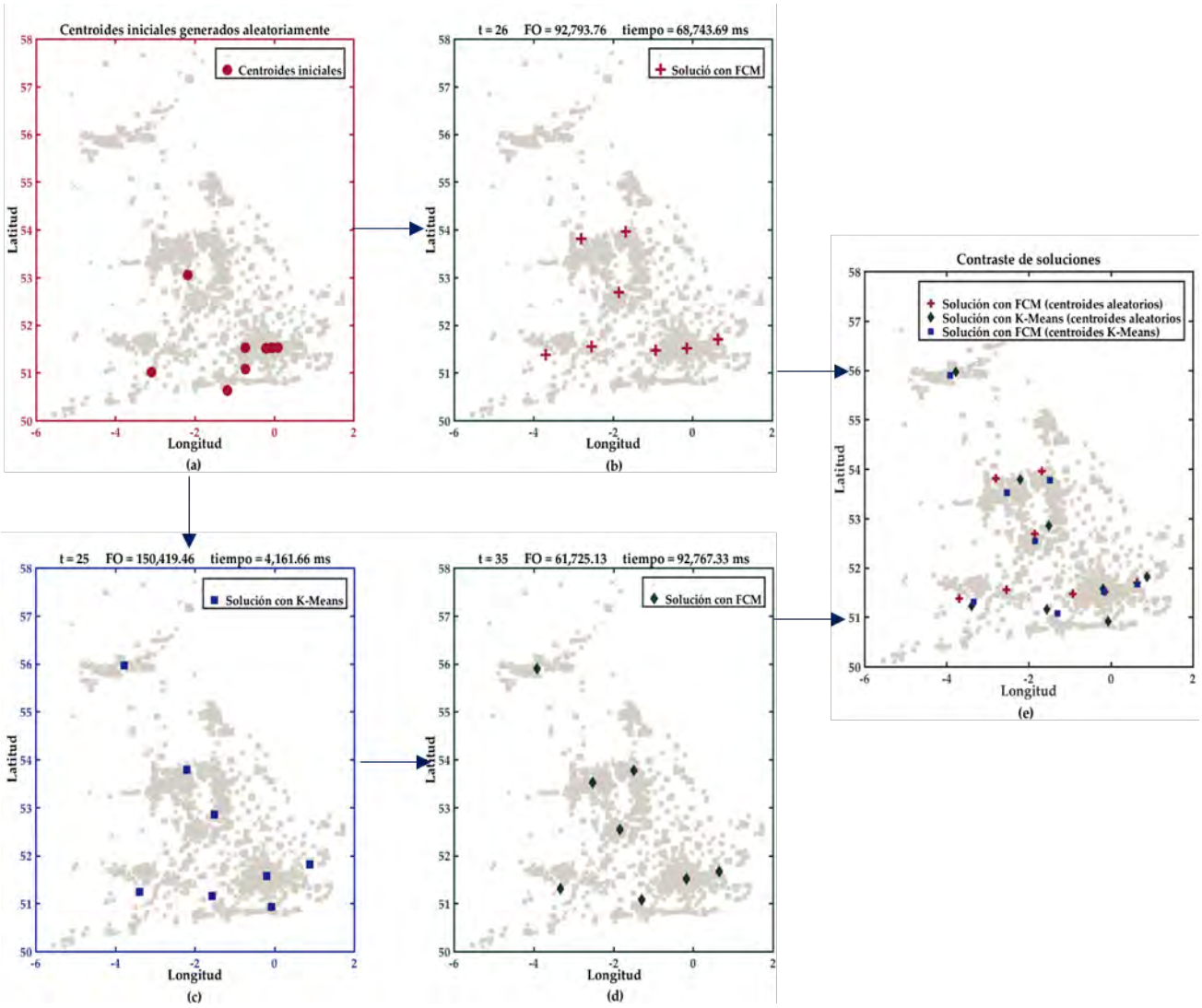

Figura 6. Centroides iniciales y finales de los algoritmos *K-Means* y *FCM* estándar con el dataset *URBAN* del Ejemplo 2

Es importante mencionar que, en los diferentes experimentos llevados a cabo, el comportamiento del algoritmo *K-Means* fue similar al Ejemplo 1. Es decir, en la mayoría de las veces, los centroides finales se acercan a la vecindad de los centroides finales del *FCM*. No obstante, en algunas ocasiones no sucede de esta manera.

3.3 Observaciones del análisis de las principales mejoras

En la literatura, hasta la fecha, se han encontrado mejoras en las fases de inicialización y clasificación del algoritmo *FCM*. En el Capítulo 2, se presentaron tres mejoras en la fase de inicialización [37, 62, 64]. Estas mejoras muestran que es factible utilizar al algoritmo *K-Means* estándar y algunas de sus variantes para utilizar los centroides de salida de dichos algoritmos y pasarlos como parámetro de entrada al *FCM*. En este sentido, se identifican las siguientes observaciones con respecto a las mejoras del algoritmo en la fase de inicialización:

Observación 1. Los centroides que genera el algoritmo *K-Means* son utilizados como parámetro de entrada al algoritmo *FCM*.

Observación 2. Los centroides del algoritmo *K++* son utilizados como parámetros de entrada para el algoritmo *FCM*.

Observación 3. Los centroides de una variante del *K++* son utilizados como parámetros de entrada al algoritmo *FCM*.

De las observaciones expuestas, se concluye que es factible utilizar algunas de las variantes del algoritmo *K-Means* para reducir la complejidad temporal del algoritmo *FCM*.

3.3.1 Selección de variantes del algoritmo *K-Means*

El algoritmo *FCM* tiene mayor complejidad computacional respecto al algoritmo *K-Means* [82-84]. En la literatura, se identificaron investigaciones que aprovechan esta ventaja del algoritmo

K-Means para iniciar los centroides del algoritmo *FCM* [37, 62, 64]. Para esta investigación, se seleccionaron los algoritmos *K++* [63] y *O-K-Means* [85]. Estos algoritmos han mostrado obtener resultados prometedores cuando se resuelven grandes *datasets* como los que se presentan en *Big Data*. A continuación, se describen detalladamente cada uno de estos algoritmos.

Algoritmo *K*++

El algoritmo *K++*, propuesto en [63], inicializa los centroides del agrupamiento del algoritmo *K-Means*, seleccionando objetos del *dataset* que están más alejados entre sí de manera probabilística. Este método acelera la velocidad de convergencia, garantizándose teóricamente que es $O(\log k)$ y, por tanto, competitivo con la solución óptima. El pseudocódigo del algoritmo *K++* se muestra en el Algoritmo 3.

Algoritmo 3: *K++*

Entrada: X, k
Salida: V
1 **Inicialización:**
2 $X := \{x_1, \ldots, x_n\}$;
3 Asignar el valor para k;
4 $V := \{\}$;
5 $V := V \cup \{v_1\}$; // Seleccionar el primer centroide v_1 aleatoriamente
6 Para $i = 2$ hasta k:
7 Seleccionar el i-th centroide v_i de X con probabilidad $D(x_i, v_j)/\sum_{x \epsilon X} D(x_i, v_j)$;
8 $V := V \cup \{v_i\}$;
9 Fin del para
10 Retornar V;
11 Fin del algoritmo

En el Algoritmo 3, se muestra la inicialización de los centroides con el algoritmo *K++*. Dado un *dataset* X y el valor de k, el siguiente paso es seleccionar el primer centroide v_1 de manera aleatoriamente de X. Posteriormente, asignarlo al conjunto de centroides V. Para el segundo centroide y hasta completar el total de k, seleccionar el *i-th* centroide con una probabilidad de distribución. Para efecto de esta investigación, el algoritmo será mencionado como *K++*, cuando se trate de inicializar los centroides.

Algoritmo *O-K-Means*

Existen diferentes mejoras para el algoritmo *K-Means*. En [85], se propuso una heurística denominada *O-K-Means*, la cual acelera el proceso de convergencia, parando el algoritmo cuando el total de los objetos que cambian de grupo en una iteración es menor a un umbral. Este valor, expresa una relación entre el esfuerzo computacional y la calidad de la solución. El pseudocódigo del algoritmo *O-K-Means* se muestra en el Algoritmo 4.

Algoritmo 4: *O-K-Means*

Entrada: $X, V, k, \varepsilon ok$
Salida: V, U

Inicialización:
  $X := \{x_1, \ldots, x_n\}$;
  $V := \{v_1, \ldots, v_k\}$;
  εok := Valor del umbral para determinar la convergencia del *O-K-Means*;
Clasificación:
  Para $x_i \in X$ y $v_k \in V\{$
    Calcular la distancia Euclidiana de cada x_i a los k centroides;
    Asignar el objeto x_i al centroide v_k más cercano;
    Calcular γ $\}$;
Cálculo de centroides:
  Calcular el nuevo conjunto de centroides V;
Convergencia:
  Si ($\gamma \leq \varepsilon ok$):
    Detener el algoritmo;
  En caso contrario:
    Ir a **Clasificación**
Fin del algoritmo

El indicador γ representa el porcentaje de objetos que cambian de grupo en una iteración t. El cual se calcula de la siguiente forma: $\gamma_t = 100(o_t/n)$, donde o_t es el número de objetos que cambian de grupo. εok, es el valor del umbral para determinar la convergencia del algoritmo y su valor propuesto en esta investigación es de 0.72.

A partir de los algoritmos *K++* y *O-K-Means* se crea un algoritmo híbrido denominado *O-K-Means++*. El pseudocódigo del algoritmo *O-K-Means++* se muestra en el Algoritmo 5.

Algoritmo 5: *O-K-Means++*

Entrada: $X, k, \varepsilon ok$
Salida: V, U

Inicialización:
  $X := \{x_1, \ldots, x_n\}$;
  Asignar el valor para k;
  $V := \{\}$;
  $V := V \cup \{v_1\}$; // Seleccionar el primer centroide v_1 aleatoriamente
  Para $i = 2$ hasta k:
    Seleccionar el i-th centroide v_i de X con probabilidad $D(x_i, v_j)/\sum_{x \epsilon X} D(x_i, v_j)$;
    $V := V \cup \{v_i\}$;
  Fin del para
  Retornar V;
  $\varepsilon ok :=$ Valor del umbral para determinar la convergencia del *O-K-Means*;
Clasificación:
  Para $x_i \in X$ y $v_k \in V\{$
    Calcular la distancia Euclidiana de cada x_i a los k centroides;
    Asignar el objeto x_i al centroide v_k más cercano;
    Calcular γ };
Cálculo de centroides:
  Calcular el nuevo conjunto de centroides V;
Convergencia:
  Si ($\gamma \leq \varepsilon ok$):
    Detener el algoritmo;
  En caso contrario:
    Ir a **Clasificación**
Fin del algoritmo

3.4 Propuesta del nuevo algoritmo en la fase de inicialización

Para resolver el problema expuesto en las observaciones del apartado 3.3, *HOFC*M toma una muestra de l soluciones de un *dataset*. Adicionalmente, se utiliza el algoritmo híbrido *O-K-Means++* descrito en la Sección 3.3.1. En la Figura 7, se describe la estructura del algoritmo *HOFCM*.

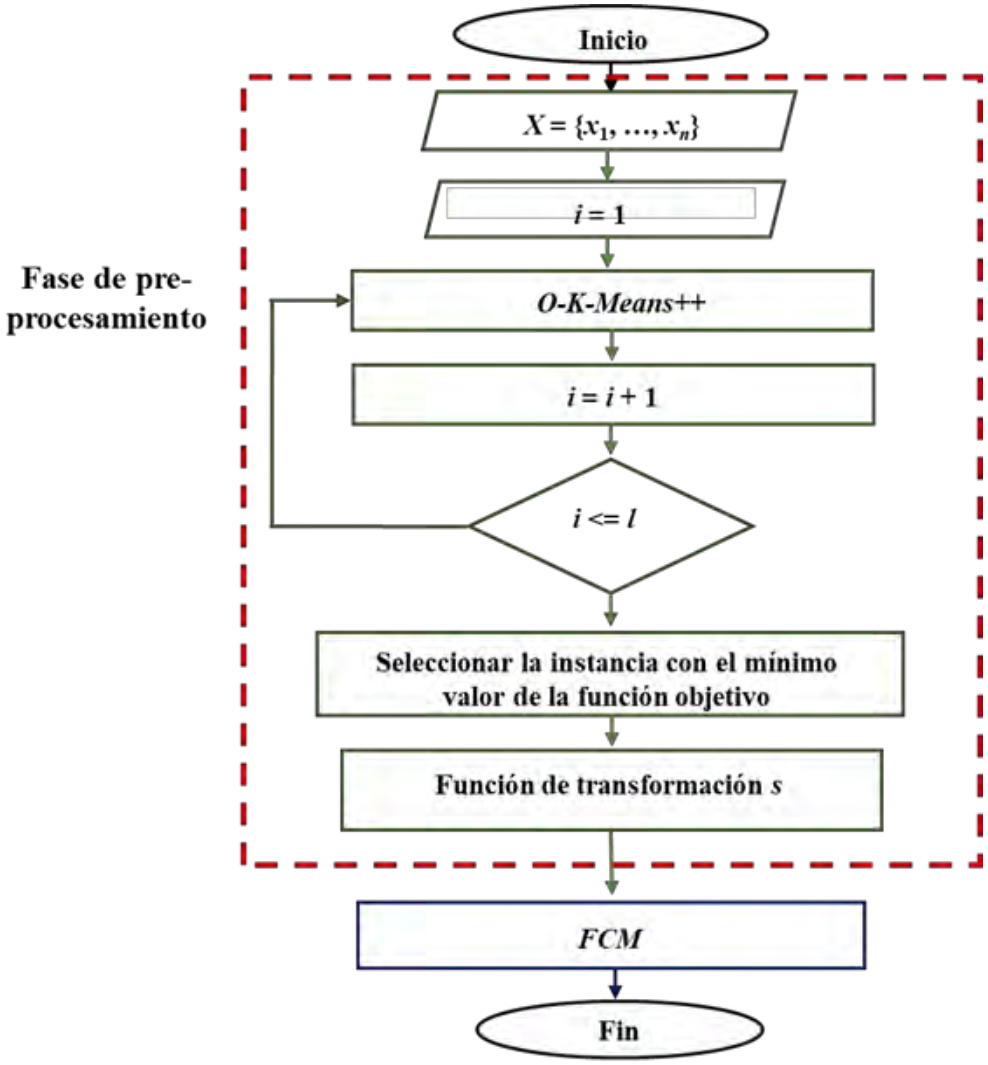

Figura 7. Estructura del algoritmo *HOFCM*

En la Figura 7, se observa que dado un *dataset X*, se cuenta con una fase de pre-procesamiento, la cual consiste en generar un conjunto de *l* soluciones de dicho *dataset*. Para ello, se ejecuta el algoritmo *O-K-Means++*, el cual devuelve centroides optimizados. Del conjunto de *l* soluciones del *dataset X*, se identifica el valor de *i* en el que la función objetivo obtuvo el mínimo valor y se seleccionan sus centroides finales. Estos centroides finales, son transformados en los valores iniciales de la matriz de pertenencia a través de una función de transformación *s*.

Se destaca que los tiempos que se generen en la fase de pre-procesamiento fueron sumados al tiempo que tarde en converger el algoritmo *FCM*. Los componentes de la función de transformación *s*, se describen con más detalle en la siguiente subsección.

Función de transformación s

La función de transformación *s*, tiene tres elementos: dominio, codominio y reglas de transformación. La siguiente notación matemática corresponde a la descripción del dominio:

$X = \{x_1, \ldots, x_n\}$ el conjunto de n objetos a particionar, donde $x_i \in \Re^d$ para $i = 1, \ldots, n$; siendo $\Re^d$ un espacio euclidiano.

$V = \{v_1, \ldots, v_c\}$ es el conjunto de centroides finales de una solución obtenida con el algoritmo *O-K-Means*, donde v_j es el centroide del grupo j.

x_i y v_j son vectores descritos en la Expresión 5 y 6.

m = valor del exponente de ponderación donde $1 < m < \infty$.

El co-dominio es representado por μ_{ij}. Donde μ_{ij} es la matriz que contiene los valores de pertenencia de cada objeto x_i a cada grupo j.

Las reglas de transformación son definidas en la Expresión 10 y 11.

$$u_{ij} = 1, \qquad \text{si } x_i = v_j \tag{10}$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\| x_i - v_j \|^2}{\| x_i - v_k \|^2} \right)^{\frac{1}{m-1}}} \quad \text{si } x_i \neq v_j \tag{11}$$

La función de transformación *s* es definida en la Expresión 12.

$$s(x_i, v_j, m) = \mu_{ij} \tag{12}$$

Llevar a cabo este algoritmo es razonable, debido a que, en la parte experimental, el algoritmo *O-K-Means* ha mostrado obtener resultados prometedores. Por esta razón, el algoritmo *FCM* tendría menor cantidad de trabajo en el agrupamiento y, por lo tanto, se reduciría el número de iteraciones. El pseudocódigo del algoritmo *HOFCM* se muestra en el Algoritmo 6.

Algoritmo 6: *HOFCM*

Entrada: *X, c, m, ε. εok*
Salida: *V, U*

```
1   Inicialización:
2       i := 1;
3       Repetir:
4           Función O-K-Means++ (X, εok, c):
5               Retorna V;
6           i = i + 1;
7       Mientras i <= l;
8       Seleccionar V para el valor de i en el que la función objetivo obtuvo el mínimo valor;
9       Función de transformación s;
10      t := 1;
11  Cálculo de centroides:
12      Calcular los centroides con Ex. (3);
13  Clasificación:
14      Actualizar y calcular la matriz de pertenencia con Ex. (4);
15  Convergencia:
16      Si max [abs(μij^(t) - μij^(t+1))] < ε:
17          Detener el algoritmo;
18      En caso contrario:
19          U^(t) := U^(t+1) y t := t + 1;
20      Ir a Cálculo de centroides
21  Fin del algoritmo
```

El algoritmo *HOFCM* tiene cuatro fases. La primera, es la fase de inicialización, en esta fase se crean dos funciones: 1) la función *O-K-Means++*, recibe como parámetros de entrada, el *dataset X*, el valor del umbral *εok* para determinar la convergencia del algoritmo y el número de

grupos, al finalizar la función, devuelve centroides optimizados definidos con la variable *V*; y 2) la función de transformación *s*, permite transformar los centroides finales de *O-K-Means++*, tomados del menor valor de la función objetivo, en los valores iniciales optimizados de la matriz de pertenencia. Los cuales son el parámetro de entrada para el algoritmo *FCM*. La segunda, tercera y cuarta fase del algoritmo *HOFCM*, denominadas cálculo de centroides, clasificación y convergencia respectivamente, son propias del *FCM* estándar.

Capítulo 4

Implementación y validación experimental de los resultados de *HOFCM*

En este Capítulo se describe la implementación del algoritmo propuesto en esta investigación, denominado *HOFCM*. Asimismo, se muestra la validación experimental en un marco metodológico de los resultados. Con este propósito, se diseñaron y ejecutaron un conjunto de experimentos con *datasets* reales y sintéticos.

4.1 Metodología

El marco metodológico que se tomó como base para desarrollar el nuevo algoritmo *HOFCM*, es el propuesto por la Dra. Catherine C. McGeoch en su libro titulado "*A guide to experimental algorithmics*" [86]. Este modelo del proceso experimental se enfoca en el desarrollo de este tipo de algoritmos. En la Figura 8, se muestra dicho proceso, en el cual los experimentos se llevan a cabo alternando dos etapas principales: Planificación y Ejecución.

En la etapa de Planificación, cuando se planifica un experimento, se parte de la formulación de una pregunta de investigación, una vez formulada, se procede al diseño de los experimentos y a la construcción del ambiente de pruebas, los cuales pueden estar comprendidos

por un programa de prueba, *datasets* de entrada, un generador de *datasets*, herramientas de medición y herramientas de visualización de datos.

En la etapa de Ejecución, se corren los experimentos. Asimismo, se almacenan y analizan los resultados. Cuando los resultados son satisfactorios generalmente se publican, en caso contrario, el proceso continúa con el fin de restructurar el diseño de experimentos o nuevamente considerar la etapa de Planificación. La naturaleza del proceso no es secuencial sino cíclica.

De este modo, con la metodología de la Dra. McGeoch se llevaron a cabo un conjunto de pruebas, las cuales se corrieron en condiciones de laboratorio y con un control de variables para evaluar el desempeño del nuevo algoritmo, principalmente tomando en cuenta la calidad del resultado y el tiempo de solución.

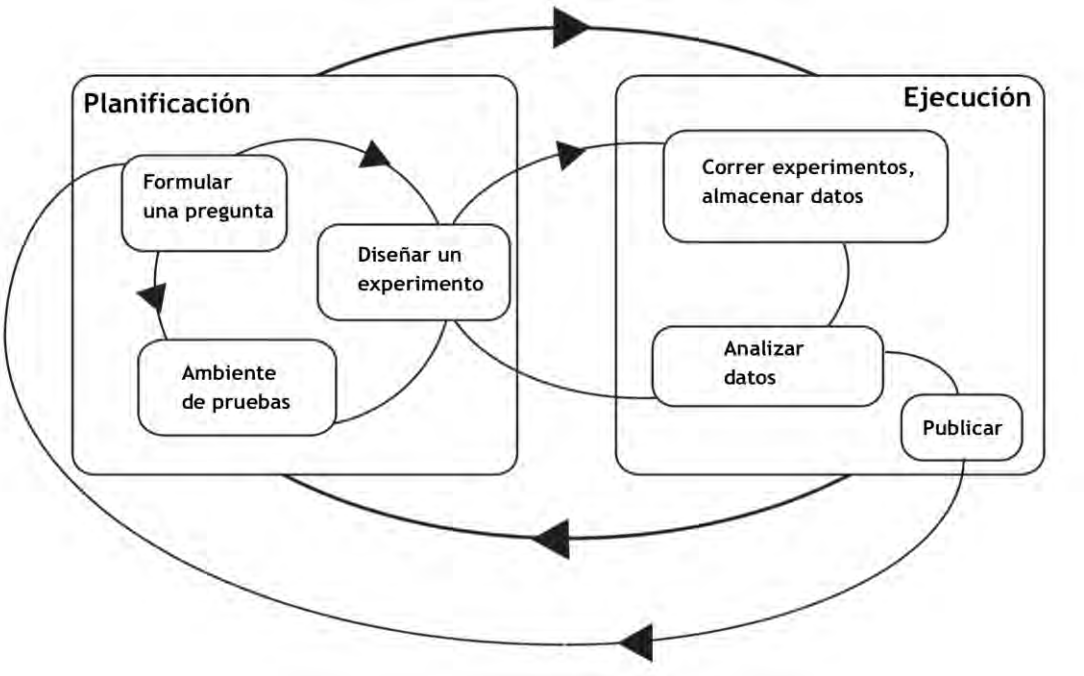

Figura 8. Proceso experimental para el análisis de algoritmos

4.2 Descripción del entorno de pruebas

Para evaluar el desempeño de *HOFCM*, se seleccionó un conjunto de cuatro *datasets* reales, obtenidas del repositorio de aprendizaje automático de la *UCI* [18]. Adicionalmente, se generó un conjunto de *datasets* sintéticos, con valores de atributos de manera aleatoria con una distribución uniforme con valores entre 0 y 1. Para contrastar los resultados de *HOFCM*, se seleccionaron e implementaron los siguientes algoritmos: *K-Means*, *O-K-Means*, *K++*, *FCM* estándar, *FCM++*, *NFCM* y *HOFCM*. Todos los algoritmos se codificaron en lenguaje *C*, con el compilador *GCC* 7.4.0. El equipo de cómputo utilizado tiene la siguiente configuración: Intel® Core™ i9-10900 2.80 GHz, 32 GB RAM, 1 TB en disco duro y un sistema operativo Windows 10 Pro.

Los algoritmos *FCM* estándar, *FCM++*, *NFCM* y *HOFCM* se ejecutaron con los valores de los parámetros: m=2, ε = 0.01. Para el *FCM* estándar se generaron de manera aleatoria los valores iniciales de la matriz de pertenencia; para el algoritmo *FCM++*, sus centroides, se generaron con el algoritmo *K++* y una variante de éste, para el algoritmo *NFCM*. En el caso del algoritmo *HOFCM*, los valores de la matriz de pertenencia, fueron obtenidos con la función de transformación *s*, descrita en la sección 3.4.

Los centroides para el algoritmo *K-Means*, fueron generados de manera aleatoria; y para el *O-K-Means*, se generaron con el algoritmo *K++*. Adicionalmente, es conveniente mencionar que el valor del umbral para determinar la convergencia del *O-K-Means* εok = 0.72.

Para todos los algoritmos implementados, el valor de c = 2, 4, 6, 8, 10, 14, 18 y 26. Con este valor, se consideran ocho configuraciones de casos de prueba para cada *dataset*, es decir, uno para cada valor de c.

El valor de l se determinó en base en pruebas preliminares en las cuales se observó que un tamaño de muestra de entre 10 y 5 ejecuciones proporcionaba buenos resultados. En este sentido, los algoritmos *FCM* estándar, *FCM++*, *NFCM* se corrieron 10 y 5 veces para cada una de las ocho configuraciones de casos de prueba de cada *dataset*.

En la Tabla 1, se describen los *datasets* utilizados. La estructura de la Tabla es la siguiente: la columna uno contiene el identificador del *dataset*; la columna dos el nombre; la columna tres el tipo de dato; las columnas cuatro y cinco contienen el total de objetos y dimensiones del *dataset*; la columna seis, el producto de las columnas cuatro y cinco. Es importante destacar que los *datasets SPAM*, *URBAN 250_2d*, *500_2d* y *1m2d*, para el caso de esta investigación, son considerados *datasets* grandes tomando en consideración el indicador *n*d* mayor a 200,000 objetos.

Tabla 1. *Datasets* utilizados en los experimentos

Id	Nombre	Tipo	*n*	*d*	Indicador de tamaño *n*d*
1	*WDBC*	Real	569	30	17,070
2	*ABALONE*	Real	4,177	7	29,239
3	*SPAM*	Real	4,601	57	262,257
4	*URBAN*	Real	360,177	2	720,354
5	*250_2d*	Sintética	250,000	2	500,000
6	*500_2d*	Sintética	500,000	2	1,000,000
7	*1m2d*	Sintética	1,000,000	2	2,000,000

4.3 Descripción de los casos de prueba

De manera general, el diseño de los experimentos se focalizó en mostrar la calidad de la solución y la eficiencia del algoritmo *HOFCM* al resolver *datasets* de gran tamaño como los utilizados en *Big Data*.

4.3.1 Descripción del Experimento I

El propósito de este primer experimento, fue contrastar los resultados de los algoritmos *FCM* estándar y *HOFCM*, considerando el entorno de pruebas descrito en la Sección 4.2 y el valor del tamaño de la muestra $l = 10$.

En la Tabla 2, se muestran los porcentajes promedio de reducción de tiempo y la ganancia de la función objetivo, resultado de la comparación de los algoritmos *FCM* estándar y *HOFCM*, para todos los *datasets* reales del Experimento I. La estructura de la Tabla 2, es la siguiente: la columna uno, contiene el número consecutivo de los casos de prueba; la columna dos, el valor de cada grupo; las columnas tres, cinco, siete y nueve, los porcentajes promedio de tiempo de solución de cada *dataset* respectivamente. Finalmente, las columnas cuatro, seis, ocho y 10, los porcentajes promedio del valor de la función objetivo para cada *dataset*. Se destacan en color rojo los valores de pérdida de tiempo y reducción en la calidad de la solución. Asimismo, se resalta en color amarillo los mejores resultados obtenidos en cada tabla.

Tabla 2. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento I

***HOFCM* versus *FCM* estándar**									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-10.46**	0.00	**-100.70**	0.00	10.90	0.00	**-120.18**	4.12
2	4	28.97	0.02	**-20.75**	0.00	44.08	6.53	53.95	0.00
3	6	19.08	0.24	5.27	0.10	89.05	22.75	60.70	0.49
4	8	31.02	**-1.19**	41.69	0.07	**89.55**	36.89	63.22	1.34
5	10	29.75	3.24	58.99	0.10	72.65	51.80	68.34	1.89
6	14	72.62	4.59	19.76	**-0.10**	86.11	47.91	70.52	1.17
7	18	28.97	13.87	**-32.28**	**-0.07**	80.42	56.83	71.86	8.39
8	26	31.10	28.48	68.29	**-3.29**	89.54	**68.12**	40.75	12.97

En las Expresiones 13 y 14, se muestran cómo se obtuvieron los porcentajes de reducción de tiempo y ganancia en la función objetivo respectivamente:

$$tiempo = 100\left(1 - \frac{tiempo\_HOFCM}{tiempoPromedio\_FCMestándar}\right) \tag{13}$$

$$Jm = 100\left(1 - \frac{Jm\_HOFCM}{JmPromedio\_FCMestándar}\right) \tag{14}$$

De la misma manera que en la Tabla 2, en la Tabla 3 se muestran los porcentajes promedio de reducción de tiempo de solución, así como la ganancia de la función objetivo, resultado de la

comparación de los algoritmos *FCM* estándar y *HOFCM*, para todos los *datasets* sintéticos del Experimento I.

Tabla 3. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento I

***HOFCM* versus *FCM* estándar**							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	65.51	20.50	56.11	20.49	**-35.10**	0.03
2	4	76.76	50.27	74.08	50.22	45.00	0.03
3	6	56.89	57.00	55.57	56.98	21.66	0.35
4	8	76.74	62.71	52.67	62.64	71.19	0.22
5	10	77.63	65.41	51.74	65.47	8.46	0.13
6	14	47.05	69.18	**87.62**	69.17	81.85	**-0.32**
7	18	79.17	71.47	74.30	71.48	76.17	0.13
8	26	82.50	**74.34**	80.49	74.32	82.62	0.01

En la Figura 9, se muestran los promedios de tiempo en milisegundos de cada uno de los ocho casos de prueba de los siete *datasets* del Experimento I.

4.3.2 Descripción del Experimento II

El propósito de este experimento, fue contrastar los resultados de los algoritmos *FCM* estándar y los de *HOFCM*, considerando el entorno de pruebas descrito en la Sección 4.2, así como el valor de la muestra $l = 5$.

De la misma manera que en el Experimento I, en las Tablas 4 y 5 se presentan los porcentajes promedio de reducción de tiempo y la ganancia de la función objetivo, resultado de la comparación de los algoritmos *FCM* estándar y *HOFCM*, para todos los *datasets* reales y sintéticos respectivamente.

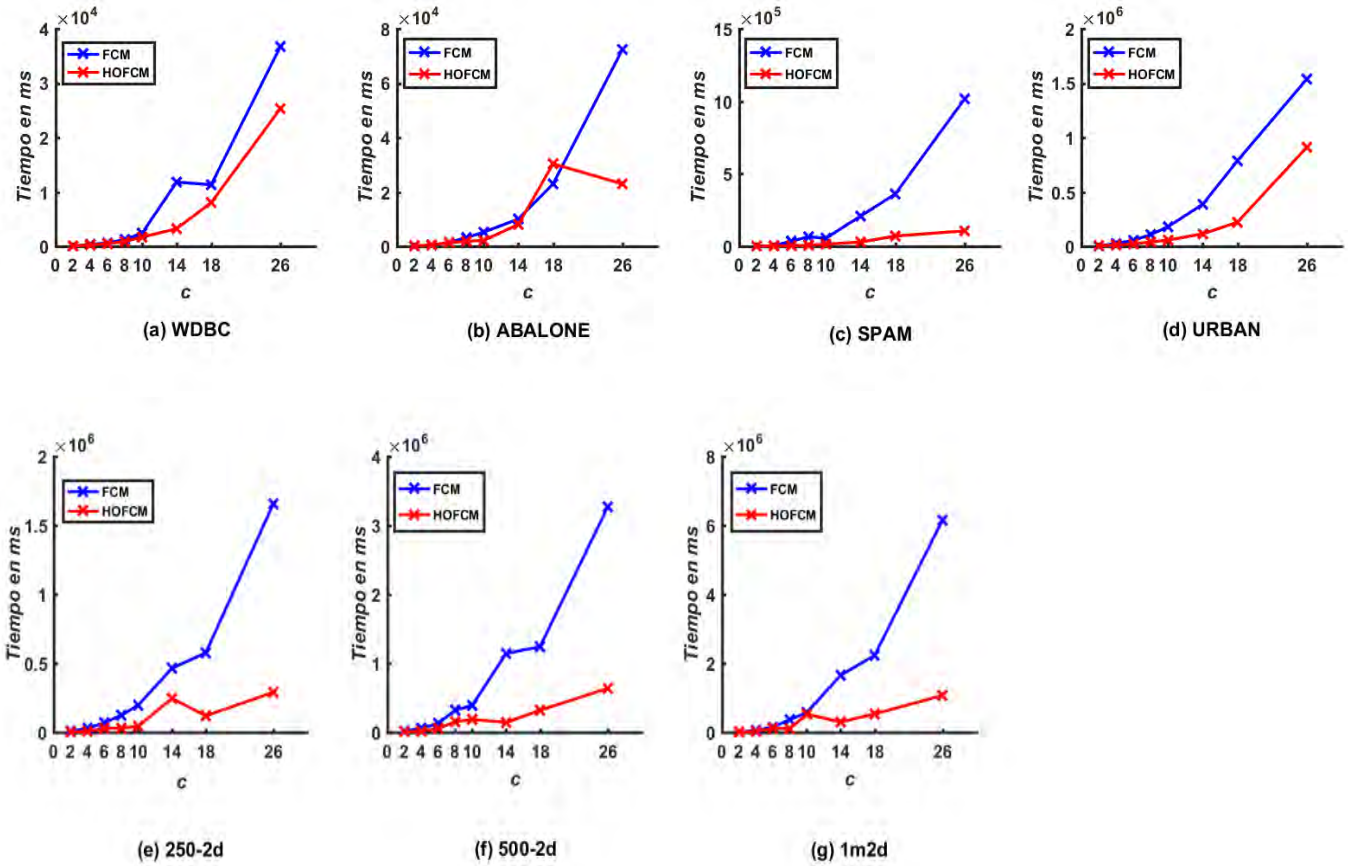

Figura 9. Resultados del agrupamiento del Experimento I

Tabla 4. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento II

***HOFCM* versus *FCM* estándar**									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	45.73	0.01	3.60	0.01	38.57	0.00	**-52.80**	7.92
2	4	68.52	0.02	23.16	**-0.02**	64.74	6.54	70.15	0.00
3	6	39.61	0.55	37.88	0.10	**92.57**	22.75	70.18	0.97
4	8	42.14	**-1.08**	60.27	0.08	92.53	36.89	75.24	1.07
5	10	**-77.43**	3.27	75.41	0.10	77.39	50.29	78.00	2.31
6	14	83.33	6.64	34.92	**-0.10**	86.88	48.06	77.47	1.59
7	18	21.21	13.77	44.70	**-0.04**	75.91	55.59	70.49	9.61
8	26	82.40	21.59	38.30	**-3.23**	90.71	**68.12**	42.27	12.99

Tabla 5. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento II

***HOFCM* versus *FCM* estándar**							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	80.48	20.50	78.51	20.49	22.51	0.02
2	4	86.64	50.27	86.56	50.22	67.98	0.03
3	6	65.87	57.00	62.70	56.99	35.74	0.27
4	8	84.09	62.71	51.27	62.65	84.73	0.00
5	10	82.50	65.41	64.22	65.46	70.30	0.01
6	14	42.47	69.33	**91.95**	69.31	86.30	**-0.38**
7	18	75.24	71.49	75.58	71.48	79.91	0.20
8	26	86.16	**74.32**	81.89	**74.32**	81.85	0.14

En la Figura 10, se muestran los promedios de tiempo en milisegundos de cada uno de los ocho casos de prueba de los siete *datasets* del Experimento II.

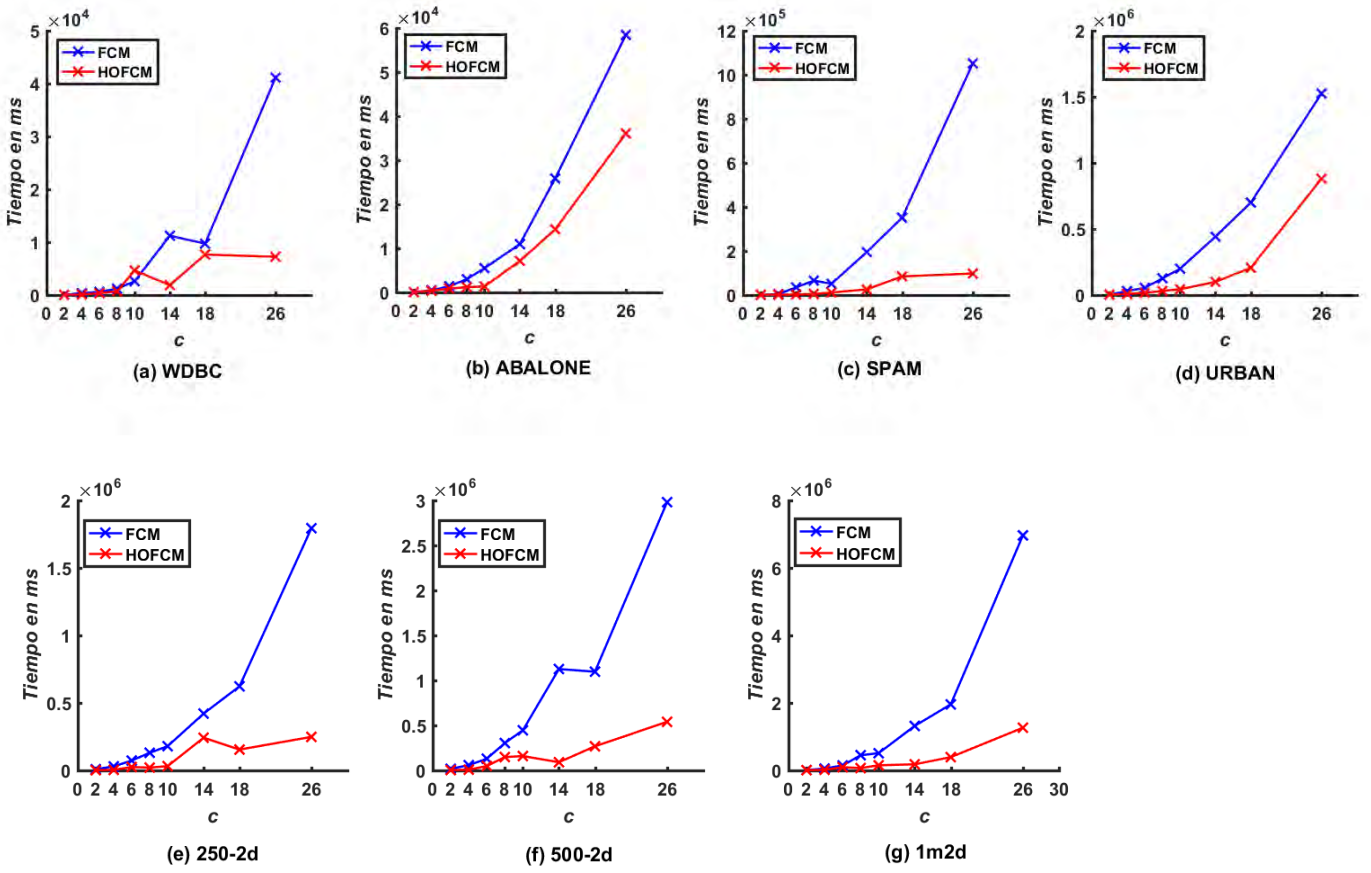

Figura 10. Resultados del agrupamiento del Experimento II

4.3.3 Descripción del Experimento III

El propósito de este experimento, fue contrastar los resultados de *HOFCM* con otras mejoras del algoritmo *FCM* estándar las cuales se denominan *FCM++* [62], *FCM-KMeans* [64] y *NFCM* [37]. El valor de la muestra $l = 10$.

En las Tablas 6 a 11, se muestran los porcentajes promedio de reducción de tiempo y la ganancia de la función objetivo, resultado de la comparación de los algoritmos *HOFCM* versus *FCM++*, *HOFCM* versus *FCM-KMeans* y *HOFCM* versus *NFCM*, respectivamente, para todos los *datasets* y casos de prueba del Experimento III. La estructura de las tablas, es similar a la Tabla 2. Se utilizan las Expresiones 13 y 14, para obtener los porcentajes de reducción de tiempo y ganancia en la función objetivo respectivamente.

Tabla 6. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los datasets reales del Experimento III, *HOFCM* versus *FCM++*

HOFCM* versus *FCM++									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-91.40**	0.00	**-127.37**	0.00	**-13.89**	0.00	**-234.32**	0.00
2	4	**-18.06**	0.02	**-74.96**	0.00	**-6.51**	0.01	7.29	0.00
3	6	23.09	1.87	8.15	0.14	32.17	**5.47**	44.21	2.26
4	8	18.58	**-0.22**	15.78	0.05	31.75	1.96	57.18	2.13
5	10	29.34	1.97	47.56	0.10	43.69	**-0.01**	59.92	1.74
6	14	29.58	0.52	22.30	0.05	54.66	0.11	41.68	**-1.06**
7	18	**-9.22**	1.87	**-45.78**	0.47	34.81	2.28	46.89	4.18
8	26	**-42.54**	4.48	69.15	**-1.93**	**59.92**	1.66	7.68	3.11

Tabla 7. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento III, *HOFCM* versus *FCM++*

HOFCM* versus *FCM++							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-233.52**	0.10	**-336.63**	0.07	**-266.33**	0.03
2	4	**-15.23**	0.02	**-17.09**	0.03	**-19.07**	0.03
3	6	**-9.83**	0.12	**-43.11**	0.15	10.95	**-0.02**
4	8	61.21	0.02	20.29	0.00	68.79	0.01
5	10	63.43	**-0.07**	20.17	**0.27**	**-7.31**	0.11
6	14	11.50	0.11	75.20	**-0.10**	**76.50**	**-0.24**
7	18	66.41	**-0.12**	58.02	0.06	70.20	0.06
8	26	74.55	0.14	71.88	0.25	75.37	0.07

Tabla 8. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento III, *HOFCM* versus *FCM-KMeans*

HOFCM* versus *FCM-KMeans									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-361.89**	0.00	**-372.86**	0.00	**-28.97**	0.00	**-185.46**	0.00
2	4	**-320.58**	0.00	**-148.64**	0.03	4.91	6.53	**-39.02**	0.00
3	6	**-70.31**	**-0.07**	**-121.73**	0.10	**86.55**	22.75	61.17	1.38
4	8	**-0.72**	**-1.30**	**-113.25**	**-0.02**	52.71	14.64	40.63	0.06
5	10	43.83	3.30	33.58	0.13	**-16.70**	35.15	45.33	3.60
6	14	79.32	4.61	32.28	0.71	74.22	48.77	63.63	**-0.99**
7	18	72.61	13.48	**-2.72**	0.47	74.69	58.18	48.78	9.97
8	26	62.56	30.15	70.41	**-2.03**	88.26	**66.67**	**-0.62**	6.39

Tabla 9. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento III, *HOFCM* versus *FCM-KMeans*

HOFCM* versus *FCM-KMeans							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-160.48**	0.05	**-210.73**	0.06	**-138.46**	0.03
2	4	**-144.28**	0.00	**-169.15**	0.00	**-135.83**	0.00
3	6	**-78.24**	0.41	**-65.63**	0.59	**-63.07**	**0.41**
4	8	**-43.73**	**-0.02**	**-116.53**	**-0.03**	8.00	**-0.03**
5	10	**-69.72**	**-0.05**	**-244.15**	0.19	**-363.06**	0.23
6	14	**-79.49**	**-0.08**	**64.68**	**-0.01**	62.94	0.01
7	18	15.37	**-0.15**	**-23.47**	0.03	12.95	0.10
8	26	13.60	0.08	19.58	0.09	21.06	**-0.02**

Tabla 10. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento III, *HOFCM* versus *NFCM*

HOFCM* versus *NFCM									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-46.42**	0.01	**-113.60**	0.01	**-2.36**	0.00	**-233.33**	0.00
2	4	**-28.49**	3.74	**-54.17**	0.00	**-0.96**	4.67	23.78	0.00
3	6	3.88	1.32	2.24	0.12	64.12	4.77	44.23	2.74
4	8	14.12	**-0.32**	24.60	0.76	**67.66**	9.05	60.46	1.73
5	10	14.84	3.41	52.34	0.47	10.72	**22.66**	52.85	3.81
6	14	16.41	1.24	41.11	0.18	51.20	10.60	58.92	**-0.77**
7	18	**-70.81**	4.28	**-41.88**	0.14	36.46	2.69	50.08	3.90
8	26	**-57.29**	6.51	62.94	**-3.26**	63.44	4.78	17.75	5.89

Tabla 11. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento III, *HOFCM* versus *NFCM*

HOFCM* versus *NFCM							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-229.48**	0.06	**-293.99**	0.06	**-332.58**	0.03
2	4	0.71	0.03	**-10.68**	0.03	19.11	0.03
3	6	**-13.55**	0.13	**-3.54**	**0.31**	1.11	0.13
4	8	67.15	0.25	**-8.93**	0.01	74.96	0.01
5	10	58.29	**-0.10**	25.11	0.14	**-15.17**	0.34
6	14	22.27	**-0.41**	**84.78**	**-0.34**	76.77	**-0.20**
7	18	74.05	**-0.07**	63.10	0.07	64.72	0.20
8	26	74.88	0.12	66.44	0.30	78.18	0.13

En la Figura 11, se muestran los promedios de tiempo en milisegundos de cada uno de los ocho casos de prueba de los siete *datasets*, resultado del Experimento III.

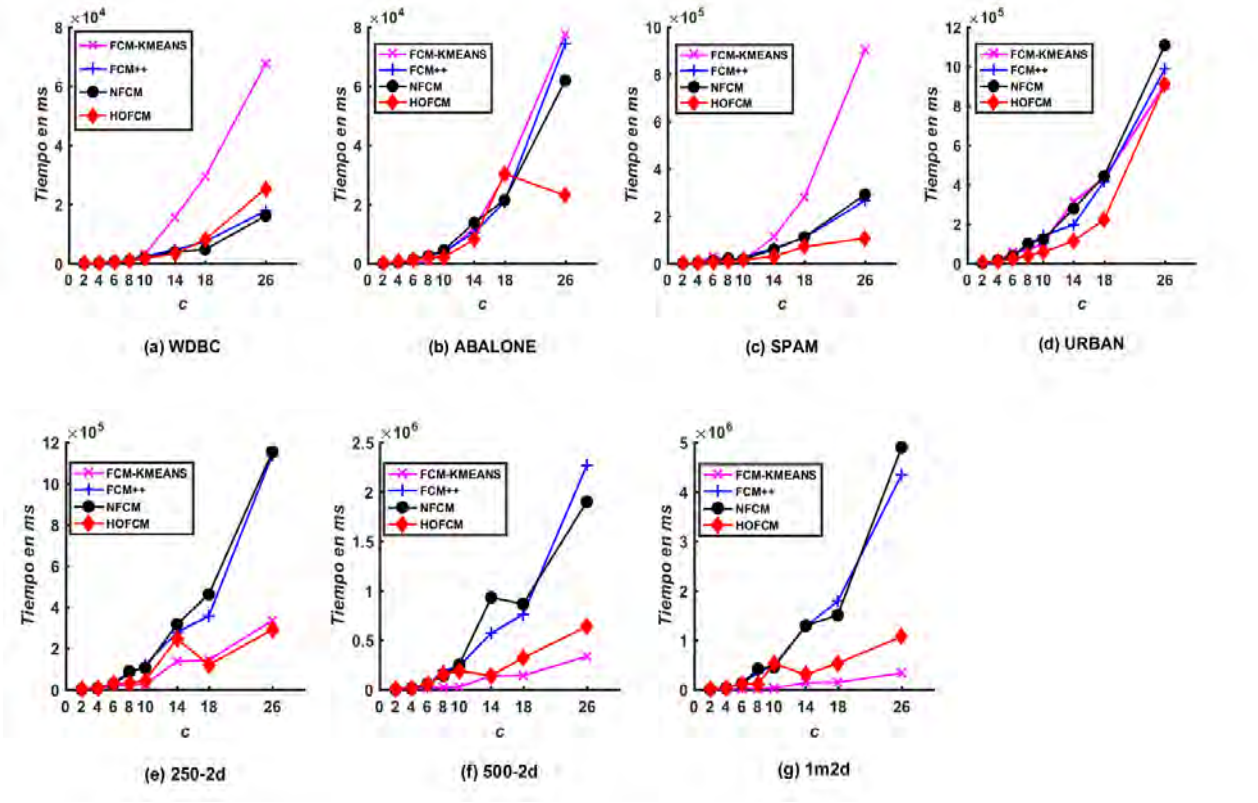

Figura 11. Resultados del agrupamiento del Experimento III

4.3.4 Descripción del Experimento IV

Con el mismo propósito del Experimento III, este experimento contrastó los resultados de *HOFCM* con *FCM++*, *FCM-KMeans* y *NFCM*. Para este experimento el valor de la muestra $l = 5$.

En las Tablas 12 a 17, se muestran los porcentajes promedio de reducción de tiempo y la ganancia de la función objetivo, resultado de la comparación entre los algoritmos *HOFCM* versus *FCM++*, *HOFCM* versus *FCM-KMeans* y *HOFCM* versus *NFCM* respectivamente, para todos los *datasets* y casos de prueba del Experimento IV.

Tabla 12. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento IV, *HOFCM* versus *FCM++*

HOFCM* versus *FCM++									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-3.82**	0.00	**-12.38**	0.00	21.68	0.00	**-114.89**	0.00
2	4	42.29	0.02	**-6.28**	**-0.02**	36.15	0.01	50.65	0.00
3	6	41.05	1.24	50.60	0.15	53.91	**7.31**	**65.25**	2.74
4	8	35.53	**-0.12**	47.00	0.06	44.09	0.76	60.08	3.54
5	10	**-49.32**	2.80	59.96	0.09	58.34	**-0.02**	66.24	2.15
6	14	53.18	4.07	39.33	0.07	60.08	**-0.97**	48.66	**-0.47**
7	18	0.79	2.37	26.99	0.01	35.30	0.83	58.77	5.45
8	26	44.31	**-4.30**	49.99	**-1.16**	60.22	2.14	19.39	3.19

Tabla 13. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento IV, *HOFCM* versus *FCM++*

HOFCM* versus *FCM++							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-75.92**	0.08	**-107.77**	0.06	**-104.90**	0.02
2	4	31.56	0.02	33.25	0.01	33.76	0.03
3	6	**-6.09**	**-0.05**	**-28.53**	0.30	35.11	0.00
4	8	78.40	0.02	13.73	0.00	82.93	0.01
5	10	79.26	**-0.12**	31.27	0.25	60.94	0.02
6	14	0.97	**0.71**	82.37	0.51	**88.12**	**-0.25**
7	18	61.69	**-0.05**	64.27	0.07	78.16	0.14
8	26	76.83	0.04	75.02	0.30	66.10	0.17

Tabla 14. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento IV, *HOFCM* versus *FCM-KMeans*

HOFCM* versus *FCM-KMeans									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-125.79**	0.00	**-123.66**	0.00	9.64	0.00	**-83.05**	0.00
2	4	**-79.66**	0.00	**-69.05**	0.01	39.53	6.54	17.32	0.00
3	6	**-15.84**	**-0.06**	**-57.14**	0.09	**91.02**	22.75	73.01	1.79
4	8	21.01	**-1.31**	**-36.20**	**-0.01**	65.83	14.64	49.54	0.25
5	10	**-84.83**	3.27	57.61	0.12	10.59	35.15	51.84	2.86
6	14	88.24	6.66	31.34	1.49	77.62	48.77	56.79	1.42
7	18	70.87	13.55	40.14	1.06	69.61	56.84	59.04	9.20
8	26	88.82	23.93	45.37	**-1.96**	89.33	**66.60**	11.01	6.74

Tabla 15. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento IV, *HOFCM* versus *FCM-KMeans*

HOFCM* versus *FCM-KMeans							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-47.08**	0.05	**-46.06**	0.07	**-36.60**	0.03
2	4	**-42.42**	**-0.01**	**-36.06**	0.00	**-40.32**	0.00
3	6	**-52.17**	0.25	**-42.54**	0.29	**-58.87**	**0.54**
4	8	**-10.17**	**-0.02**	**-97.55**	**-0.03**	38.59	**-0.03**
5	10	**-14.75**	**-0.07**	**-194.38**	0.22	**-39.99**	0.15
6	14	**-70.50**	0.49	72.29	0.48	**80.37**	0.00
7	18	**-1.71**	**-0.08**	**-9.45**	**-0.03**	39.05	0.13
8	26	**-20.22**	**-0.10**	34.97	0.07	-7.14	0.14

Tabla 16. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* reales del Experimento IV, *HOFCM* versus *NFCM*

HOFCM* versus *NFCM									
		WDBC		***ABALONE***		***SPAM***		***URBAN***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	36.45	0.01	**-14.14**	0.01	32.78	0.00	**-110.94**	0.00
2	4	42.48	0.02	**-12.07**	**-0.03**	41.66	5.30	53.13	0.00
3	6	14.47	**-0.08**	35.23	0.11	64.64	3.53	55.83	3.67
4	8	34.60	**-1.07**	55.49	0.09	**80.22**	11.27	58.50	1.93
5	10	**-76.73**	2.40	62.54	0.08	32.97	**24.59**	58.08	4.44
6	14	58.27	3.09	48.57	0.03	62.38	6.49	67.54	0.20
7	18	**-27.96**	2.05	35.76	0.04	31.15	0.61	59.99	4.34
8	26	55.60	-2.11	16.05	**-3.41**	59.73	7.46	15.75	7.07

Tabla 17. Porcentajes promedio de reducción de tiempo y valor de la función objetivo de los *datasets* sintéticos del Experimento IV, *HOFCM* versus *NFCM*

HOFCM* versus *NFCM							
		250_2d		***500_2d***		***1m2d***	
P	***c***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***	***% tiempo***	***% J_m***
1	2	**-88.47**	0.05	**-91.27**	0.07	**-141.09**	0.02
2	4	32.66	0.03	45.26	0.03	50.17	0.02
3	6	**-27.10**	0.10	**-5.81**	**0.62**	**-0.47**	0.25
4	8	79.25	0.48	**-14.23**	0.01	84.28	0.02
5	10	67.78	**-0.07**	36.43	0.17	65.39	0.05
6	14	25.68	0.08	**90.18**	0.02	85.98	**-0.19**
7	18	61.39	0.02	63.99	0.06	77.44	0.10
8	26	76.69	0.06	67.49	0.37	73.98	0.25

En la Figura 12, se muestran los promedios de tiempo en milisegundos, resultado de cada uno de los ocho casos de prueba de los siete *datasets* del Experimento IV.

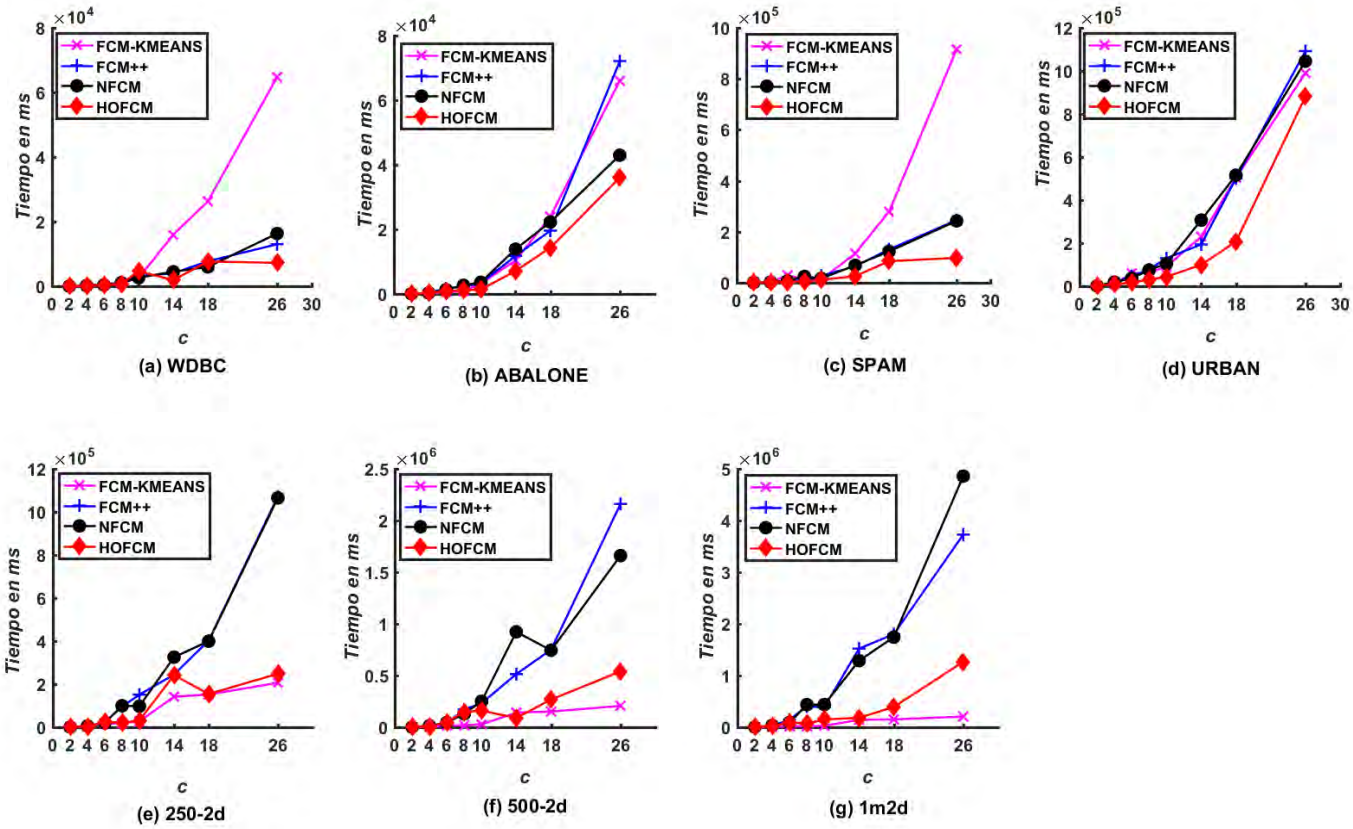

Figura 12. Resultados del agrupamiento del Experimento IV

Capítulo 5

Análisis de resultados y conclusiones

En este Capítulo se presentan: el análisis de los resultados de esta investigación, sus conclusiones y se proponen temas para trabajos futuros. Con base en los resultados de esta investigación, se muestra que el algoritmo propuesto denominado *Hybrid OK-Means Fuzzy C-Means* (*HOFCM*) reduce la complejidad temporal del algoritmo *Fuzzy C-Means* estándar mediante la optimización de la matriz de pertenencia inicial.

5.1 Análisis de experimentos

En este apartado, se analizarán los resultados obtenidos después de resolver los casos de prueba de cada uno de los *datasets* en los experimentos I, II, III y IV.

5.1.1 Análisis de resultados del Experimento I

Con base en los resultados experimentales mostrados en las Tablas 2 y 3 y en la Figura 9, en la cual se muestran los resultados de la comparación entre *HOFCM* y *FCM* estándar, se observa lo siguiente:

HOFCM, para todos los *datasets* grandes, tanto sintéticos como reales, logró reducir el tiempo para todos los casos de prueba para $c > 2$. En el mejor de ellos, logró reducir hasta un

89.55%. Este porcentaje se resalta con amarillo en la columna siete de la Tabla 2. Es de destacar, que este porcentaje se logró con el *dataset SPAM*, el cual tiene una alta dimensionalidad.

HOFCM, logró mejorar la calidad de la solución en los *datasets* reales, en 28 de 32 casos. En el mejor de los casos, se logró mejorar la calidad de la solución en 68.12%. Dicho porcentaje se resalta con amarillo en la columna ocho de la Tabla 2. En el peor de los casos, se obtuvo una pérdida de la calidad en un 3.29%, identificada en la columna seis. Con respecto a los *datasets* sintéticos, logró mejorar la calidad de la solución en 23 de 24 casos de prueba. En el mejor de los casos se logró mejorar la calidad de la solución en 74.34%, identificado en la Tabla 3. En el peor de los casos se obtuvo una pérdida de la calidad en un 0.32%.

A partir de los alentadores resultados obtenidos, se está en condiciones de afirmar que la propuesta *HOFCM* presentó mejor desempeño que el algoritmo *FCM* estándar. Es importante destacar, que el tiempo de solución del algoritmo *HOFCM*, se suma a las 10 soluciones muestra de la fase de pre-procesamiento del algoritmo *O-K-Means++*.

5.1.2 Análisis de resultados del Experimento II

Con base en los resultados experimentales mostrados en las Tablas 4 y 5, así como en la Figura 10, en las cuales se muestran los resultados de *HOFCM* y *FCM* estándar, se observa lo siguiente:

HOFCM, para todos los *datasets* grandes, tanto sintéticos como reales, logró reducir el tiempo en todos los casos de prueba para $c > 2$. En el mejor de ellos, logró reducir hasta un 92.57%. Este porcentaje se resalta con amarillo en la columna siete de la Tabla 4. Es importante destacar, que este porcentaje continuó siendo el *dataset SPAM*, al igual que en el Experimento I. Adicionalmente, el tiempo de solución para los *datasets* pequeños (*WDBC* y *ABALONE*), mejoró con respecto al Experimento I.

HOFCM, logró mejorar la calidad de la solución en los *datasets* reales, en 27 de 32 casos. En el mejor de los casos, se logró mejorar la calidad de la solución en 68.12%. Dicho porcentaje se resalta con amarillo en la columna ocho de la Tabla 4. En el peor de los casos, se obtuvo una pérdida de la calidad en un 3.23%, identificada en la columna seis. Con respecto a los *datasets* sintéticos, logró mejorar la calidad de la solución en 23 de 24 casos de prueba. En el mejor de los casos se logró mejorar la calidad de la solución en 74.32%, identificado en la Tabla 5. En el peor de los casos se obtuvo una pérdida de la calidad en un 0.38%.

De acuerdo con los resultados obtenidos, se está en condiciones de afirmar que la propuesta *HOFCM* tiene mejor desempeño que el algoritmo *FCM* estándar. Es importante enfatizar que el tiempo de solución del algoritmo *HOFCM*, se suma a las cinco soluciones muestra de la fase de pre-procesamiento del algoritmo *O-K-Means++*. Se recomienda tener un valor de $l = 5$ para los *datasets* grandes y pequeños ya que de manera general se tiene mayor reducción de tiempo. Lo anterior, se puede observar en la Figura 10.

5.1.3 Análisis de resultados del Experimento III

Considerando los resultados de las Tablas 6 a 11 y la Figura 11, en las cuales se presentan los resultados de *HOFCM* y otros algoritmos como *FCM++*, *FCM-KMeans* y *NFCM*, se observa lo siguiente:

HOFCM superó al algoritmo *FCM++*, en el tiempo de solución, en todos los casos de prueba con los *datasets* grandes, reales y sintéticos para $c > 6$. En el mejor de los casos, el tiempo de solución fue de 59.92% para los *datasets* reales y de 76.50% para los sintéticos, respectivamente (ver Tablas 6 y 7). En los *datasets* pequeños, en 9 de 16 casos de prueba *HOFCM* fue superior. Respecto a la calidad de la solución, en los *datasets* grandes, fue superior en 33 de los 40 casos de prueba. En el mejor de los casos se obtuvo una ganancia en la calidad de la solución de 5.47%, identificado en la Tabla 6.

HOFCM superó al algoritmo *FCM-KMeans* solamente en 26 de los 56 casos de prueba. Se destaca que con el *dataset SPAM*, *HOFCM* mejoró los tiempos de solución en seis de los ocho

casos de prueba, su mejor porcentaje de tiempo fue de 88.55%. Nótese que, con respecto a la calidad de la solución, con este *dataset*, *HOFCM* fue superior en todos los casos, mejorando la calidad hasta en un 66.67%. En general, en 43 de 56 casos fue superior en la calidad de la solución.

HOFCM superó al algoritmo *NFCM*, en el tiempo de solución, en todos los casos con los *datasets* grandes y reales para c > 4 y c > 10 para los datasets sintéticos. En el mejor de los casos, para los *datasets* reales, se obtuvo un porcentaje de 67.66%, identificado en la Tabla 10. Para los sintéticos un porcentaje de 84.78%, identificado en la Tabla 11. Respecto a la calidad de la solución superó en 29 de 32 casos de prueba con los *datasets* reales. En el mejor de los casos se logró un 22.66% y en el peor de los casos un 3.26%.

En las tres comparaciones se observa que *HOFCM* obtuvo mayores reducciones de tiempo y ganancias en la calidad de solución que los otros algoritmos, cuando se aplica *datasets* grandes y reales.

5.1.4 Análisis de resultados del Experimento IV

Considerando los resultados expuestos en las Tablas 12 a 17 y en la Figura 12, en las que se presentan los resultados de *HOFCM* y otros algoritmos como *FCM++*, *FCM-KMeans* y *NFCM*, para $l = 5$ se observa lo siguiente:

HOFCM superó al algoritmo *FCM++*, en el tiempo de solución, en todos los casos de prueba con los *datasets* grandes y reales para $c > 2$. En el mejor de los casos, el tiempo de solución fue de 65.25% para los *datasets* reales. Con respecto a los sintéticos, fue superior para $c > 6$, logrando un porcentaje de reducción de 88.12% para los sintéticos, respectivamente (ver Tablas 12 y 13). En los *datasets* pequeños, en 12 de 16 casos de prueba *HOFCM* fue superior. Respecto a la calidad de la solución, en los *datasets* grandes, fue superior en 33 de los 40 casos de prueba. En el mejor de los casos se obtuvo una ganancia en la calidad de la solución de 7.31%, identificado en la Tabla 12.

HOFCM superó al algoritmo *FCM-KMeans* en todos los casos de prueba para los datasets grandes y reales para c > 2. Se destaca que con el *dataset SPAM*, *HOFCM* mejoró los tiempos de solución en ocho de los ocho casos de prueba, su mejor porcentaje de tiempo fue de 91.02%. Nótese que, con respecto a la calidad de la solución, superó en todos los casos de prueba, en el mejor de los casos logró un 66.60% (ver Tabla 14).

HOFCM superó al algoritmo *NFCM*, en el tiempo de solución, en todos los casos con los *datasets* grandes y reales para c > 2 y c > 10 para los datasets sintéticos. En el mejor de los casos, para los *datasets* reales, se logró un porcentaje de 80.22% identificado en la Tabla 16. Para los sintéticos un porcentaje de 90.18%, identificado en la Tabla 17. Respecto a la calidad de la solución, superó en 28 de 32 casos de prueba con los *datasets* reales. En el mejor de los casos se logró un 24.59% y en el peor de los casos un 3.41%.

En las tres comparaciones, se observa que *HOFCM* obtuvo mayores reducciones de tiempo, así como mayores ganancias en la calidad de solución, cuando se trata de *datasets* grandes y reales. Se recomienda tener un valor de $l = 5$ para los *datasets* grandes y reales ya que de manera general se tiene mayor reducción de tiempo. Lo anterior, se puede observar en la Figura 12.

En la Tabla 18, se muestra la sumatoria de los tiempos promedio de solución en milisegundos, para los ocho casos de prueba de cada *dataset* y de cada algoritmo implementado en los cuatro experimentos realizados dentro de esta investigación para $l = 10$. La estructura de la Tabla 18 es la siguiente: en la columna uno se muestra el nombre del *dataset*; en las columnas dos a seis, el nombre de cada algoritmo y en la columna siete, el porcentaje promedio de reducción de tiempo del algoritmo *HOFCM* vs *FCM*.

En la Tabla 18, se observa que el algoritmo *FCM* estándar es el que genera más tiempo en la convergencia de los ocho casos de prueba para todos los *datasets*. *HOFCM*, superó en todos los casos de prueba el porcentaje de reducción de tiempo de solución con respecto al *FCM*

estándar. En el mejor de los casos, se obtuvo una reducción del 85.55%. De manera general se puede afirmar que el algoritmo *HOFCM* es más rápido en 3.86, 1.32, 2.54 y 2.66 veces que los algoritmos *FCM* estándar, *FCM-KMeans*, *FCM++* y *NFCM* respectivamente.

Tabla 18. Sumatoria de los tiempos de solución en milisegundos de los cuatro experimentos para $l = 10$

	Algoritmos					
Nombre del *dataset*	***FCM* estándar**	***FCM-KMeans***	***FCM++***	***NFCM***	***HOFCM***	**% Reducción de tiempo *HOFCM* vs *FCM***
WDBC	64,516.85	116,784.57	34,037.94	28,395.16	39,925.41	38.12
ABALONE	115,702.20	123,930.39	113,680.29	105,730.07	67,382.41	41.76
SPAM	1,746,301.34	1,355,655.16	482,874.71	513,205.42	234,909.71	**86.55**
URBAN	3,080,009.70	1,886,664.03	1,885,697.60	2,104,360.89	1,381,155.78	55.16
250_2d	3,123,721.19	680,754.03	2,001,945.65	2,159,213.21	768,099.50	75.41
500_2d	6,570,274.37	1,619,756.55	4,079,241.19	4,155,819.90	1,519,170.28	76.88
1m2d	11,181,675.94	3,091,877.12	8,385,299.13	8,729,514.14	2,688,326.24	75.96
Tiempo total de solución	25,882,201.59	8,875,421.85	16,982,776.51	17,796,238.80	6,698,969.33	74.12
Número de veces que el algoritmo *HOFCM* supera a:	**3.86**	**1.32**	**2.54**	**2.66**		

En la Tabla 19, se muestra la sumatoria de los tiempos promedio de solución en milisegundos, para los ocho casos de prueba de cada *dataset* y cada algoritmo implementado en los cuatro experimentos realizados en esta investigación para $l = 5$. La estructura de la Tabla 19, es la misma que la Tabla 18.

Tabla 19. Sumatoria de los tiempos de solución en milisegundos de los cuatro experimentos para $l = 5$

	Algoritmos					
Nombre del *dataset*	***FCM* estándar**	***FCM-KMeans***	***FCM++***	***NFCM***	***HOFCM***	**% Reducción de tiempo *HOFCM* vs *FCM***
WDBC	66,853.65	110,705.25	29,710.75	31,098.69	22,591.95	66.21
ABALONE	105,716.97	105,164.76	111,240.17	87,108.36	61,343.54	41.97
SPAM	1,753,977.24	1,367,874.00	485,627.75	486,269.41	228,991.46	**86.94**
URBAN	3,077,824.31	1,949,316.25	2,060,909.80	2,104,490.21	1,289,889.07	58.09
250_2d	3,257,707.82	566,429.90	1,995,794.90	2,013,608.89	726,984.24	77.68
500_2d	6,158,175.70	1,573,156.25	3,878,928.95	3,769,289.22	1,265,632.16	79.45
1m2d	11,407,827.98	3,041,541.14	8,013,107.88	8,894,520.56	2,176,925.03	80.92
Tiempo total de solución	25,828,083.66	8,714,187.54	16,575,320.20	17,386,385.36	5,772,357.44	77.65
Número de veces que el algoritmo *HOFCM* supera a:	**4.47**	**1.51**	**2.87**	**3.01**		

En la Tabla 19, se observa que el algoritmo *FCM* estándar es el que genera más tiempo en la convergencia de los ocho casos de prueba para todos los *datasets*. *HOFCM*, superó en todos los casos de prueba el porcentaje promedio de reducción de tiempo de solución con respecto al *FCM* estándar. En el mejor de los casos, se obtuvo una reducción del 86.94%. De manera general, se concluye que el algoritmo *HOFCM* es más rápido en 4.47, 1.51, 2.87 y 3.01 veces que los algoritmos *FCM* estándar, *FCM-KMeans*, *FCM++* y *NFCM* respectivamente.

5.2 Conclusiones

El algoritmo *HOFCM* reduce el número de iteraciones y por lo tanto el tiempo de solución cuando se resuelven grandes *datasets*, como los que se presentan en *Big Data*. Se destaca, en la mayoría de los casos de prueba, una ganancia en la calidad de la solución.

Las principales contribuciones de la presente investigación son las siguientes:

1. La inicialización del algoritmo *FCM* con los centroides finales del *K-Means*, muestra una buena estrategia, lo cual es factible debido a que la complejidad temporal del algoritmo *K-Means* es menor que la del algoritmo *FCM*. Además, los centroides finales del *K-Means*, en la mayoría de los casos, están cerca de la vecindad de los centroides finales del algoritmo *FCM*.

2. La estrategia anterior, se logra perfecciona al seleccionar dos mejoras del algoritmo *K-Means*. La primera denominada *O-K-Means*, la cual acelera el proceso de convergencia, parando el algoritmo cuando el total de los objetos que cambian de grupo en una iteración es menor a un umbral. La segunda denominada *K++*, la cual inicializa los centroides de la siguiente manera: el primero de manera aleatoria y a partir del segundo centroide seleccionando objetos del *dataset* que están más alejados entre sí de manera probabilística. Con ambas mejoras se crea un algoritmo híbrido denominado *O-K-Means++*.

3. El algoritmo híbrido *O-K-Means++* tiene una naturaleza aleatoria y no siempre proporciona buenos resultados. Una manera de resolver este problema es crear una muestra de *n* soluciones. La técnica de muestreo propuesta es a través de ejecutar *n* soluciones de un *dataset* con el algoritmo *O-K-Means++*. El mejor conjunto de centroides, se selecciona considerando el mínimo valor de la función objetivo. Estos centroides finales, son transformados en los valores iniciales de la matriz de pertenencia a través de una función de transformación *s*, para posteriormente ejecutar el algoritmo

FCM estándar. De manera experimental, el tamaño de la muestra más adecuado fue de cinco.

Para validar los resultados del algoritmo *HOFCM*, se diseñaron y ejecutaron un conjunto de experimentos compuestos de conjuntos de datos reales y sintéticos. Para contrastar los resultados de *HOFCM*, fueron seleccionados e implementados los siguientes algoritmos: *FCM* estándar, *FCM-KMeans*, *FCM++* y *NFCM*. Con base en los resultados experimentales, se observó que *HOFCM* obtuvo una reducción en el tiempo de solución en todos los *datasets* grandes, comparado con el algoritmo *FCM* estándar. Al contrastar los resultados de *HOFCM* con los algoritmos *FCM-KMeans*, *FCM++* y *NFCM*, se observó mayor rapidez promedio en 1.51, 2.87 y 3.01 veces respectivamente. En particular, se observó que el algoritmo *HOFCM* obtuvo un buen desempeño en la solución de los *datasets* grandes. Por ejemplo, en el *dataset SPAM*, el cual tiene una alta dimensionalidad, redujo el tiempo en un 92.57%, comparándolo con el *FCM* estándar. Con respecto a la calidad de la solución, se observó que en la mayoría de los casos de prueba se logró una ganancia. En el mejor de los casos se logró una ganancia de 68.12% con los *datasets* reales y un 74.34% con los *datasets* sintéticos. En el peor de los casos se redujo la calidad en un 3.29%.

Con base en los resultados de esta investigación, se afirma que *HOFCM* resuelve *datasets* más grandes en menor tiempo que el algoritmo *FCM* estándar, mejorando la calidad de la solución en la mayoría de los casos. *HOFCM* no brinda beneficios en *datasets* pequeños, por ejemplo, *datasets* del tamaño del conjunto de datos *IRIS*. Por lo anterior, el algoritmo *HOFCM* es recomendable para los usuarios que priorizan la reducción del tiempo de solución. En particular, es útil cuando se solucionan *datasets* como los que se presentan en *Big Data*.

Finalmente, se considera que los principios utilizados en nuestro enfoque podrían ser utilizados en otras variantes que mejoren el algoritmo *FCM*.

5.3 Investigaciones futuras

Con el fin de dar continuidad a la investigación que se reporta en esta tesis, se sugiere continuar con los siguientes tópicos que podrán aportar a la heurística *HOFCM*:

a) Implementar *HOFCM* bajo el paradigma de programación paralela y/o distribuido.
b) Estudiar teórica y experimentalmente las relaciones entre *HOFCM* y el conjunto de casos que ha resuelto mejor.

Referencias

[1] Statista Research Departmen. "Volume of Data/Information Created, Captured, Copied, and Consumed Worldwide from 2010 to 2020, with Forecasts from 2021 to 2025". Accessed: Feb. 10, 2023 [Online.] Available: https://www.statista.com/statistics/871513/worldwide-data-created/

[2] J. M. Cebrian, B. Imbernón, J. Soto, and J. M. Cecilia, "Evaluation of Clustering Algorithms on HPC Platforms," *Math.,* vol. 9, no. 17, pp. 1-20, Sep. 2021, doi:10.3390/math9172156.

[3] N. Radwan, "Big Data Ethics," *Int. J. Comput. Sci. Inf. Secur. (IJCSIS),* vol. 19, no. 1, pp. 80-85, Jan. 2021, doi.org/10.5281/zenodo.4533716.

[4] Z. Dafir, Y. Lamari, and S. C. Slaoui, "A survey on parallel clustering algorithms for Big Data," *Artif. Intell. Rev.,* vol. 54, no. 4, pp. 2411-2443, Apr. 2021, doi.org/10.1007/s10462-020-09918-2.

[5] M. A. Mahdi, K. M. Hosny, and I. Elhenawy, "Scalable Clustering Algorithms for Big Data: A Review," *IEEE Access,* vol. 9, pp. 80015-80027, Jun. 2021, doi.org/10.1109/ACCESS.2021.3084057.

[6] T. H. Sardar and Z. Ansari, "MapReduce-based Fuzzy C-means Algorithm for Distributed Document Clustering," *J. Inst. Eng. India Ser. B,* vol. 103, no. 1, pp. 131-142, Feb. 2022, doi:10.1007/s40031-021-00651-0.

[7] P. Giordani, M. B. Ferraro, and F. Martella, "Big Data and Clustering," in *An Introduction to Clustering with R,* Singapore: Springer Singapore, 2020, pp. 111-121.

[8] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *J. Parallel Distrib. Comput.,* vol. 74, no. 7, pp. 2561-2573, Jan, 2014, doi.org/10.1016/j.jpdc.2014.01.003.

[9] A. S. Shirkhorshidi, S. Aghabozorgi, T. Y. Wah, and T. Herawan, "Big Data Clustering: A Review," in *Comput. Sci. Appl. – ICCSA 2014*, Guimaráes, Portugal, 2014, pp. 707-720, doi.org/10.1007/978-3-319-09156-3_49.

[10] V. W. Ajin and L. D. Kumar, "Big data and clustering algorithms," in *2016 Int. Conf. Res. Advances Integr. Navigation Syst. (RAINS)*, Bangalore, India, 2016, pp. 1-5, doi:10.1109/RAINS.2016.7764405.

[11] M. S. Yang, "A survey of fuzzy clustering," *Math. Comput. Modelling,* vol. 18, no. 11, pp. 1-16, Dec. 1993, doi.org/10.1016/0895-7177(93)90202-A.

[12] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition. I," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.,* vol. 29, no. 6, pp. 778-785, Dec. 1999, doi:10.1109/3477.809032.

[13] X. Rui and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.,* vol. 16, no. 3, pp. 645-678, May. 2005, doi:10.1109/TNN.2005.845141.

[14] A. Fahad *et al.*, "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis," *IEEE Trans. Emerg. Topics Comput.,* vol. 2, no. 3, pp. 267-279, Jun. 2014, doi:10.1109/TETC.2014.2330519.

[15] J. Nayak, B. Naik, and H. S. Behera, "Fuzzy C-Means (FCM) Clustering Algorithm: A Decade Review from 2000 to 2014," in *Comput. Intell. Data Mining*, vol. 2, New Delhi, India, 2014, pp. 133-149, doi.org/10.1007/978-81-322-2208-8_14.

[16] A. K. Shukla and P. K. Muhuri, "Big-data clustering with interval type-2 fuzzy uncertainty modeling in gene expression datasets," *Eng. Appl. Artif. Intell.,* vol. 77, pp. 268-282, Jan. 2019, doi.org/10.1016/j.engappai.2018.09.002.

[17] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York, NY, USA: Plenum Press, 1981.

[18] D. Dua and C. Graff, February 2, 2023. "UCI Machine Learning Repository," distribuye by University of California, Irvine, School of Information and Computer Sciences, http://archive.ics.uci.edu/ml

[19] A. E. Ezugwu *et al.*, "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects," *Eng. Appl. Artif. Intell.,* vol. 110, pp. 1-43, Feb. 2022, doi.org/10.1016/j.engappai.2022.104743.

[20] S. Miyamoto, H. Ichihashi, and K. Honda, *Algorithms for Fuzzy Clustering Methods in c-Means Clustering with Applications* (Studies in Fuzziness and Soft Compuitng). Berlin, Germany: Springer, 2008.

[21] I. Atiyah, A. Mohammadpour, and S. Taheri, "KC-Means: A Fast Fuzzy Clustering," *Hindawi Advances Fuzzy Syst.,* vol. 2018, pp. 1-8, Jun. 2018, doi.org/10.1155/2018/2634861.

[22] J. C. Bezdek, *Elementary Cluster Analysis: Four Basic Methods that (Usually) Work*. Gistrup, Denmark: River Publishers, 2022.

[23] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statis. Probability*, Berkeley, CA, USA, 1967, pp. 281-297.

[24] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.,* vol. 14, pp. 1-37, Dec. 2007, doi:10.1007/s10115-007-0114-2.

[25] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.,* vol. 31, no. 8, pp. 651-666, Jun. 2010, doi.org/10.1016/j.patrec.2009.09.011.

[26] L. Nigro, "Performance of Parallel K-Means Algorithms in Java," *Algorithms,* vol. 15, no. 4, pp. 1-15, Mar. 2022, doi.org/10.3390/a15040117.

[27] M. Nazari, A. Hussain, and P. Musilek, "Applications of Clustering Methods for Different Aspects of Electric Vehicles," *Electron.,* vol. 12, no. 4, pp. 1-23, Feb. 2023, doi.org/10.3390/electronics12040790.

[28] S. Z. Selim and M. A. Ismail, "K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. PAMI-6, no. 1, pp. 81-87, Jan 1984, doi.org/10.1109/TPAMI.1984.4767478.

[29] L. A. Zadeh, "Fuzzy sets," *Inf. Control,* vol. 8, no. 3, pp. 338-353, Jun. 1965, doi.org/10.1016/S0019-9958(65)90241-X.

[30] R. Bellman, R. Kalaba, and L. Zadeh, "Abstraction and pattern classification," *J. Math. Anal. Appl.,* vol. 13, no. 1, pp. 1-7, Jan. 1966, doi.org/10.1016/0022-247X(66)90071-0.

[31] E. H. Ruspini, "A new approach to clustering," *Inf. Control,* vol. 15, no. 1, pp. 22-32, Jul. 1969, doi.org/10.1016/S0019-9958(69)90591-9.

[32] E. H. Ruspini, "Numerical methods for fuzzy clustering," *Inf. Sciences,* vol. 2, no. 3, pp. 319-350, Jul. 1970, doi.org/10.1016/S0020-0255(70)80056-1.

[33] J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *J. Cybern.,* vol. 3, no. 3, pp. 32-57, Sep. 1973, doi.org/10.1080/01969727308546046.

[34] M. Hashemzadeh, A. G. Oskouei, and N. Farajzadeh, "New fuzzy C-means clustering method based on feature-weight and cluster-weight learning," *Appl. Soft. Comput.,* vol. 78, pp. 324-345, May. 2019, doi.org/10.1016/j.asoc.2019.02.038.

[35] E. H. Ruspini, J. C. Bezdek, and J. M. Keller, "Fuzzy Clustering: A Historical Perspective," *IEEE Comput. Intell. Mag.,* vol. 14, no. 1, pp. 45-55, Feb. 2019, doi.org/10.1109/MCI.2018.2881643.

[36] X. Song, M. Shi, J. Wu, and W. Sun, "A new fuzzy c-means clustering-based time series segmentation approach and its application on tunnel boring machine analysis," *Mech. Syst. Signal Process.,* vol. 133, pp. 1-18, Nov. 2019, doi.org/10.1016/j.ymssp.2019.106279.

[37] Q. Liu, J. Liu, M. Li, and Y. Zhou, "Approximation algorithms for fuzzy C-means problem based on seeding method," *Theor. Comput. Sci.,* vol. 885, no. 11, pp. 146-158, Sep. 2021, doi.org/10.1016/j.tcs.2021.06.035.

[38] F. Gamino-Sánchez *et al.*, "Block-Matching Fuzzy C-Means clustering algorithm for segmentation of color images degraded with Gaussian noise," *Eng. Appl. Artif. Intell. ,* vol. 73, pp. 31-49, Aug. 2018, doi.org/10.1016/j.engappai.2018.04.026.

[39] C. Singh and A. Bala, "A transform-based fast fuzzy C-means approach for high brain MRI segmentation accuracy," *Appl. Soft Comput. ,* vol. 76, pp. 156-173, Mar. 2019, doi.org/10.1016/j.asoc.2018.12.005.

[40] P. Kaur, "Intuitionistic fuzzy sets based credibilistic fuzzy C-means clustering for medical image segmentation," *Inter. J. Infor. Technol.,* vol. 9, Sep. 2017, doi.org/10.1007/s41870-017-0039-2.

[41] S. Ghosh and S. Kumar, "Comparative Analysis of K-Means and Fuzzy C-Means Algorithms," *Int. J. Adv. Comput. Sci. Appl.,* vol. 4, no. 4, pp. 35-39, May. 2013, doi.org/10.14569/IJACSA.2013.040406.

[42] D. Bora and D. Gupta, "A Comparative study Between Fuzzy Clustering Algorithm and Hard Clustering Algorithm," *Int. J. Comp. Trends Technol.,* vol. 10, no. 2, pp. 108-113, Apr. 2014, doi.org/10.14445/22312803/IJCTT-V10P119.

[43] Z. Cebeci and F. Yıldız, "Comparison of K-Means and Fuzzy C-Means Algorithms on Different Cluster Structures," *J. Agricultural Inform.,* vol. 6, no. 3, pp. 13-23, Oct. 2015, doi.org/10.17700/jai.2015.6.3.196.

[44] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1979.

[45] G. H. Ball and D. J. Hall, "A clustering technique for summarizing multivariate data," *Behav. sci.,* vol. 12, no. 2, pp. 153-155, Mar. 1967, doi.org/10.1002/bs.3830120210.

[46] J. C. Bezdek, "Fuzzy mathematics in pattern classification," Ph.D. dissertation, Faculty of the Graduate School, Cornell Univ. Ithaca, New York, NY, USA, 1973.

[47] J. C. Bezdek, "Numerical taxonomy with fuzzy sets," *J. Math. Biol.,* vol. 1, no. 1, pp. 57-71, May. 1974, doi.org/10.1007/BF02339490.

[48] J. C. Bezdek, "Cluster Validity with Fuzzy Sets," *J. Cybern.,* vol. 3, no. 3, pp. 58-73, Feb. 1974, doi.org/10.1080/01969727308546047.

[49] J. C. Bezdek and J. Dunn, "Optimal Fuzzy Partitions: A Heuristic for Estimating the Parameters in a Mixture of Normal Distributions," *IEEE Trans. Comput.,* vol. 24, pp. 835-838, Aug. 1975, doi.org/10.1109/T-C.1975.224317.

[50] J. C. Bezdek, "Feature selection for binary data: medical diagnosis with fuzzy sets," in *AFIPS '76: Proc. June 7-10, 1976, nat. comput. conf. expo.*, New York, NY, USA, 1976, pp. 1057–1068, doi.org/10.1145/1499799.1499946.

[51] J. C. Bezdek and J. Douglas Harris, "Fuzzy partitions and relations; an axiomatic basis for clustering," *Fuzzy Sets Syst.,* vol. 1, no. 2, pp. 111-127, Apr. 1978, doi.org/10.1016/0165-0114(78)90012-X.

[52] J. C. Bezdek and J. D. Harris, "Convex decompositions of fuzzy partitions," *J. Mat. Anal. Appl.,* vol. , no. 2, pp. 490-512, Feb. 1979, doi.org/10.1016/0022-247X(79)90039-8.

[53] J. C. Bezdek, "A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. PAMI-2, no. 1, pp. 1-8, Jan. 1980, doi.org/10.1109/TPAMI.1980.4766964.

[54] W. I. Zangwill, *Nonlinear Programming: A Unified Approach*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1969.

[55] J. C. Bezdek, R. Ehrlich, and W. E. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. & Geosci.,* vol. 10, no. 2-3, pp. 191-203, Jan. 1984, doi.org/10.1016/0098-3004(84)90020-7.

[56] R. L. Cannon, J. V. Dave, and J. C. Bezdek, "Efficient Implementation of the Fuzzy c-Means Clustering Algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. PAMI-8, no. 2, pp. 248-255, Mar. 1986, doi.org/10.1109/TPAMI.1986.4767778.

[57] R. Hathaway and J. Bezdek, "Recent convergence results for the fuzzy C-means clustering algorithms," *J. Classification,* vol. 5, pp. 237-247, Sep. 1988, doi.org/10.1007/BF01897166.

[58] R. J. Hathaway and J. C. Bezdek, "Optimization of clustering criteria by reformulation," *IEEE Trans. Fuzzy Syst.,* vol. 3, no. 2, pp. 241-245, May. 1995, doi.org/10.1109/91.388178.

[59] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy c-means model," *IEEE Trans. Fuzzy Syst.,* vol. 3, no. 3, pp. 370-379, Aug. 1995, doi.org/10.1109/91.413225.

[60] R. J. Hathaway, J. C. Bezdek, and H. Yingkang, "Generalized fuzzy c-means clustering strategies using L/sub p/ norm distances," *IEEE Trans. Fuzzy Syst.,* vol. 8, no. 5, pp. 576-582, Oct. 2000, doi.org/10.1109/91.873580.

[61] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE Trans. Fuzzy Syst.,* vol. 13, no. 4, pp. 517-530, Aug. 2005*,* doi.org/10.1109/TFUZZ.2004.840099.

[62] A. Stetco, X. J. Zeng, and J. Keane, "Fuzzy C-means++: Fuzzy C-means with effective seeding initialization," *Expert Syst. Appl.,* vol. 42, no. 21, pp. 7541-7548, Nov. 2015*,* doi.org/10.1016/j.eswa.2015.05.014.

[63] D. Arthur and S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," in *SODA '07: Proc. 18th. Annu. ACM-SIAM Symp. Discrete Algorithms*, Lousiana, LA, USA, 2007, pp. 1027-1035*,* doi:10.5555/1283383.1283494.

[64] Z. Wu, G. Chen, and J. Yao, "The Stock Classification Based on Entropy Weight Method and Improved Fuzzy C-means Algorithm," in *ICBDC '19: Proc. 4th Int. Conf. Big Data Comput.*, Guangzhou, China, 2019, pp. 130-134*,* doi.org/10.1145/3335484.3335503.

[65] J. F. Kolen and T. Hutcheson, "Reducing the time complexity of the fuzzy c-means algorithm," *IEEE Trans. Fuzzy Syst.,* vol. 10, no. 2, pp. 263-267, Apr. 2002*,* doi.org/10.1109/91.995126.

[66] M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty, "A modified Fuzzy C- Mean algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Med. Imag.,* vol. 21, no. 3, pp. 193-9, Mar. 2002*,* doi.org/10.1109/42.996338.

[67] X. Wang, Y. Wang, and L. Wang, "Improving fuzzy c-means clustering based on feature-weight learning," *Pattern Recognit. Lett.,* vol. 25, no. 10, pp. 1123-1132, Jul. 2004*,* doi.org/10.1016/j.patrec.2004.03.008.

[68] W. Cai, S. Chen, and D. Zhang, "Fast and robust fuzzy c-means clustering algorithms incorporating local information for image segmentation," *Pattern Recognit.,* vol. 40, no. 3, pp. 825-838, Mar. 2007*,* doi.org/10.1016/j.patcog.2006.07.011.

[69] A. Manacero, E. Guariglia, T. A. de Souza, R. S. Lobato, and R. Spolon, "Parallel fuzzy minimals on GPU," *Appl. Sci.,* vol. 12, no. 5, p. 2385, Feb. 2022*,* doi.org/10.3390/app12052385.

[70] Q. Zhang, Z. Chen, and Y. Leng, "Distributed fuzzy c-means algorithms for big sensor data based on cloud computing," *Int. J. Sensor Netw.,* vol. 18, p. 32, Jun. 2015*,* doi.org/10.1504/IJSNET.2015.069871.

[71] J. Qin, W. Fu, H. Gao, and W. X. Zheng, "Distributed k-Means Algorithm and Fuzzy c-Means Algorithm for Sensor Networks Based on Multiagent Consensus Theory," *IEEE Trans. Cybern.,* vol. 47, pp. 1-12, Mar. 2016*,* doi.org/10.1109/TCYB.2016.2526683.

[72] J. M. Cecilia, J.-C. Cano, J. Morales-García, A. Llanes, and B. Imbernón, "Evaluation of Clustering Algorithms on GPU-Based Edge Computing Platforms," *Sensors,* vol. 20, no. 21, p. 6335, Nov. 2020*,* doi.org/10.3390/s20216335.

[73] N. A. Ali, A. E. abbassi, and B. Cherradi, "The performances of iterative type-2 fuzzy C-mean on GPU for image segmentation," *J Supercomputing,* vol. 78, no. 2, pp. 1583-1601, Jun. 2022*,* doi.org/10.1007/s11227-021-03928-9.

[74] B. Liu, S. He, D. He, Y. Zhang, and M. Guizani, "A Spark-based Parallel Fuzzy C-means Segmentation Algorithm for Agricultural Image Big Data," *IEEE Access,* vol. 7, pp. 42169-42180, Mar. 2019, doi.org/10.1109/ACCESS.2019.2907573.

[75] Y. Ma and W. Cheng, "Optimization and Parallelization of Fuzzy Clustering Algorithm Based on the Improved Kmeans++ Clustering," in *IOP Conf Ser: Mater. Sci. Eng.*, vol. 768, 2020, pp. 1-7, doi.org/10.1088/1757-899X/768/7/072106.

[76] Q. Yu and Z. Ding, "An improved Fuzzy C-Means algorithm based on MapReduce," in *2015 8th Int. Conf. Biomedical Eng. Infor. (BMEI)*, Shenyang, China, 2015, pp. 634-638, doi.org/10.1109/BMEI.2015.7401581.

[77] W. Dai, C. Yu, and Z. Jiang, "An Improved Hybrid Canopy-Fuzzy C-Means Clustering Algorithm Based on MapReduce Model," *J. Comput. Sci. Eng.,* vol. 10, no. 1, pp. 1-8, Mar. 2016, doi.org/10.5626/JCSE.2016.10.1.1.

[78] A. Almomany, A. Jarrah, and A. H. Al Assaf, "FCM Clustering Approach Optimization Using Parallel High-Speed Intel FPGA Technology," *J. Elect. Comput. Eng.,* vol. 2022, pp. 1-11, May. 2022, doi.org/10.1155/2022/8260283.

[79] O. Sakarya, "Applying fuzzy clustering method to color image segmentation," in *2015 Federated Conf. Comput. Sci. Inf. Syst. (FedCSIS)*, Lodz, Poland, 2015, pp. 1049-1054, doi.org/10.15439/2015F222.

[80] V. V. Vela-Rincón, D. Mújica-Vargas, and J. de Jesus Rubio, "Parallel hesitant fuzzy C-means algorithm to image segmentation," *Signal, Image Video Process.,* vol. 16, no. 1, pp. 73-81, Jun. 2021, doi.org/10.1007/s11760-021-01957-8.

[81] J. Pérez-Ortega *et al.*, "POFCM: A Parallel Fuzzy Clustering Algorithm for Large Datasets," *Math.,* vol. 11, no. 8, pp. 1-16, Apr. 2023, doi.org/10.3390/math11081920.

[82] Purnawansyah, Haviluddin, A. F. O. Gafar, and I. Tahyudin, "Comparison Between K-Means and Fuzzy C-Means Clustering in Network Traffic Activities," in *Proc. 11th. Int. Conf. Manage. Sci. Eng. Manage.*, Kanazawa, Japan, 2018, pp. 300-310, doi.org/10.1007/978-3-319-59280-0.

[83] I.-D. Borlea, R.-E. Precup, A.-B. Borlea, and D. Iercan, "A Unified Form of Fuzzy C-Means and K-Means algorithms and its Partitional Implementation," *Knowl.-Based Syst.,* vol. 214, p. 106731, Feb. 2021, doi.org/10.1016/j.knosys.2020.106731.

[84] X. Ran, X. Zhou, M. Lei, W. Tepsan, and W. Deng, "A Novel K-Means Clustering Algorithm with a Noise Algorithm for Capturing Urban Hotspots," *Appl. Sci.,* vol. 11, no. 23, pp. 1-21, Nov. 2021, doi.org/10.3390/app112311202.

[85] J. Pérez-Ortega, N. N. Almanza-Ortega, and D. Romero, "Balancing effort and benefit of K-means clustering algorithms in Big Data realms," (in eng), *PLoS One,* vol. 13, no. 9, pp. 1-19, Sep. 2018, doi.org/10.1371/journal.pone.0201874.

[86] C. C. McGeoch, *A Guide to Experimental Algorithmics*. Cambridge, U.K.: Cambridge Univ. Press, 2012.