Prototipo de software para la agrupación de datos con una red neuronal artificial con topología de Kohonen UDCluster

Software prototype for data grouping using a neuronal artificial net with the Kohonen topology UDCluster

Sandra del Pilar Bautista Morales
Tecnóloga en Sistematización de Datos Universidad Distrital Francisco José de Caldas.
sandris_1683@yahoo.com

Myriam Andrea Reyes Ortiz
Tecnóloga en Sistematización de Datos Universidad Distrital Francisco José de Caldas.
andreyes20@yahoo.com

Jorge Enrique Rodríguez Rodríguez
Ingeniero de Sistemas, Especialista en Diseño y Construcción de Soluciones Telemáticas, Especialista en Ingeniería de Software Universidad Distrital Francisco José de Caldas y candidato a Magister en Ingeniería de Sistemas Universidad Nacional de Colombia. Profesor Universidad Distrital Francisco José de Caldas adscrito a la Facultad Tecnológica.
jrodri@udistrital.edu.co

Fecha de recepción: septiembre 09 de 2004
Clasificación del artículo: Investigación
Fecha de aceptación: diciembre 20 de 2004

Palabras clave: Red neuronal, agrupamiento de datos, vecindad, topología Kohonen.
Key words: Neuronal net, data grouping, neighborhood, Kohonen topology.

Resumen

En este artículo se presenta el desarrollo de la herramienta UDCluster, la cual hace parte del proyecto de investigación "Desarrollo de herramientas para la minería de datos UDMiner". Primero se realiza una introducción acerca de la importancia de la agrupación (comúnmente llamada *clustering*) presentada como una técnica de la minería de datos; luego se aborda el porqué del desarrollo de esta herramienta, se hace referencia al método utilizado para la agrupación (redes neuronales con topología Kohonen) y al lenguaje de modelado y de programación empleados en el desarrollo; posteriormente, se muestran los resultados obtenidos en la solución de un problema típico de agrupación de datos y se exponen las principales conclusiones de la investigación.

Abstract

In this article, the development of the UDCluster tool is presented which makes part of the research project "Development of tools for data mining "*UD Miner*". First an introduction is presented about the importance of the grouping (commonly called

clustering) a technique of the data mining; subsequently, it is explained why the development of this tool and a reference is made to the method used for the clustering (neuronal nets with Kohonen topology), the modeling and programming language employed in the development; then the results obtained are shown by solving a typical problem of data clustering and presenting the main conclusions of the research.

1. Introducción

Una de las áreas de estudio de la minería de datos (*data mining*) desarrolla herramientas que permiten la extracción de información oculta en grandes cantidades de datos, y su análisis y evaluación; ellas son útiles para predecir comportamientos y tendencias futuras. La técnica es aplicada a fin de descubrir patrones, relaciones, grupos, reglas, asociaciones o incluso excepciones que faciliten la toma de decisiones.

UDCluster hace parte del proyecto de investigación "Desarrollo de herramientas para la minería de datos UDMiner"; se trata de una herramienta destinada a la clasificación, predicción, asociación y, especialmente, a la agrupación de datos. Mediante la técnica de agrupación, los datos son clasificados en diversos grupos (*clusters*); los elementos de un mismo grupo conservan similaridad, diferenciándose así de los demás. Así, esta técnica puede ser usada como una herramienta estándar que facilita la percepción de la distribución de los datos, para observar las características de cada grupo y agruparlos dentro de una colección particular de elementos, permitiendo así su análisis acertado. Alternativamente, la agrupación puede ser utilizada como fase de preprocesamiento para otras técnicas de minería de datos como la caracterización y clasificación, las cuales podrían operar sobre los grupos descubiertos (Han, 2001).

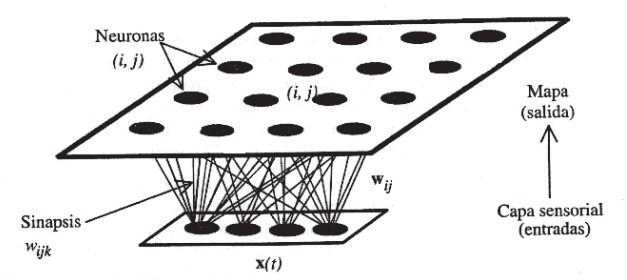
Para llevar a cabo la agrupación de los datos existe variedad de métodos: jerárquicos, basados en densidades, en particiones y en modelos, entre otros. El proceso también puede llevarse a cabo mediante la implementación de redes neuronales artificiales, que es precisamente el método utilizado en el desarrollo de esta herramienta.

Una red neuronal es un sistema de análisis de información compuesto por gran número de elementos de procesamiento, conectados entre sí a través de canales de comunicación normalmente unidireccionales, que operan sobre información local, interna o externa. Este tipo de redes se ha utilizado para reconocer y clasificar patrones, completar una señal a partir de valores parciales o reconstruir el patrón correcto partiendo de uno distorsionado, entre otras áreas; de esta forma, se constituyen en una herramienta fundamental para llevar a cabo la agrupación de datos, debido a su capacidad para modelar datos complejos y multidimensionales.

Para el desarrollo de esta aplicación se seleccionó como topología de red neuronal la de mapas autoorganizativos, o modelo de Kohonen; su algoritmo fue utilizado para ejecutar el proceso de agrupación los datos; esta topología se presentó en 1982 como sistema con un comportamiento semejante al del cerebro. Se trataba de un modelo de red neuronal con capacidad para formar mapas de características de manera similar a lo que ocurre en el cerebro; en este último hay neuronas que se organizan en muchas zonas, de forma que la información de entorno, captada a través de los órganos sensoriales, se representa internamente en forma de mapas bidimensionales (Hilera, 1996).

Los mapas autoorganizativos de Kohonen son un algoritmo, a veces clasificado dentro de las redes neuronales, que a partir de un proceso de entrenamiento agrupa los datos de acuerdo con una serie de características. La arquitectura típica de este tipo de mapas se muestra en la figura 1.

Figura 1. Arquitectura típica de los mapas de Kohonen



Como puede apreciarse, se trata de una red de tipo unidireccional organizada en dos capas: la primera se encuentra formada por las neuronas de entrada, la segunda consiste en un *array* de neuronas de dos dimensiones. Dado que se necesitan dos índices para etiquetar cada neurona, los pesos sinápticos asociados a cada neurona tendrán tres índices (i, j, k); i, j, que indican la posición de la neurona en la capa, y k, la componente o conexión con cierta neurona de entrada.

2. ¿Por qué UDCluster?

Las tecnologías de información se constituyen en componentes fundamentales de la infraestructura de las grandes organizaciones y permiten registrar múltiples detalles de la vida empresarial. El almacenamiento de cada transacción refleja la interacción de la organización con otras compañías, clientes o terceros, y su constante utilización crea la necesidad de emplear sistemas de registro potentes y eficientes que les permitan estudiarlas. Para una compañía es vital establecer estrategias y tomar decisiones que impulsen su crecimiento con base en la agrupación y análisis de la información que maneja; para realizar actividades de mercadeo la población objeto se subdivide de acuerdo con ciertas variables de interés: es el proceso de segmentación (Berry, 1997). No obstante, un inconveniente frecuente es la complejidad de los datos derivada de su gran volumen; esto hace que su estudio resulte dispendios; asimismo, datos relevantes en los análisis pueden ser descartados porque no se identifica su presencia.

La agrupación es una de las pocas actividades de la minería de datos que puede ser descrita como descubrimiento de conocimiento. En ella no hay datos preclasificados ni distinción entre variables dependientes e independientes; en su lugar se buscan grupos de datos (*clusters*) cuyos elementos guarden similitud entre sí; al final, estos grupos podrán representar clientes, proveedores o productos que la compañía maneje en áreas similares (Berry, 1997).

UDCluster implementa un método de agrupación para grandes cantidades de información, explorando la profundidad total de los datos y generando grupos a partir de su estudio. Además, mediante una interfaz amigable y de fácil manejo permite visualizar los resultados o grupos generados para que puedan ser interpretados y evaluados por el usuario para la toma de decisiones. La herramienta puede ser empleada en diversos campos: mercadotecnia, diseño, ingeniería civil, análisis de inversiones, reconocimiento de patrones, etc.

En este artículo se muestra la aplicación de la herramienta en un archivo de datos que contiene información de imágenes captadas desde un satélite. El entrenamiento y agrupación de los datos están basado en el algoritmo de Kohonen, el cual maneja un aprendizaje de tipo no supervisado[1], por lo que solo requiere información de entrada para ajustar los pesos de las conexiones entre los datos y agrupar. Cada neurona en la capa de salida representa una categoría; por tanto, solo queda un patrón representativo de cada una, que agrupa todas las propiedades de los datos contenidos en ella. La estructura de Kohonen es de red monocapa[2]; por esto, el proceso de entrenamiento puede ser más rápido en

1 El aprendizaje no supervisado se presenta cuando a la red se le proporciona solo una información de entrada; con ella la red ajusta sus interconexiones segúnsus estímulos y la salida de la propia red.

2 Una red monocapa está formada únicamente por una capa de entrada y una capa de salida; se diferencia de una red multicapa, que además posee capas ocultas.

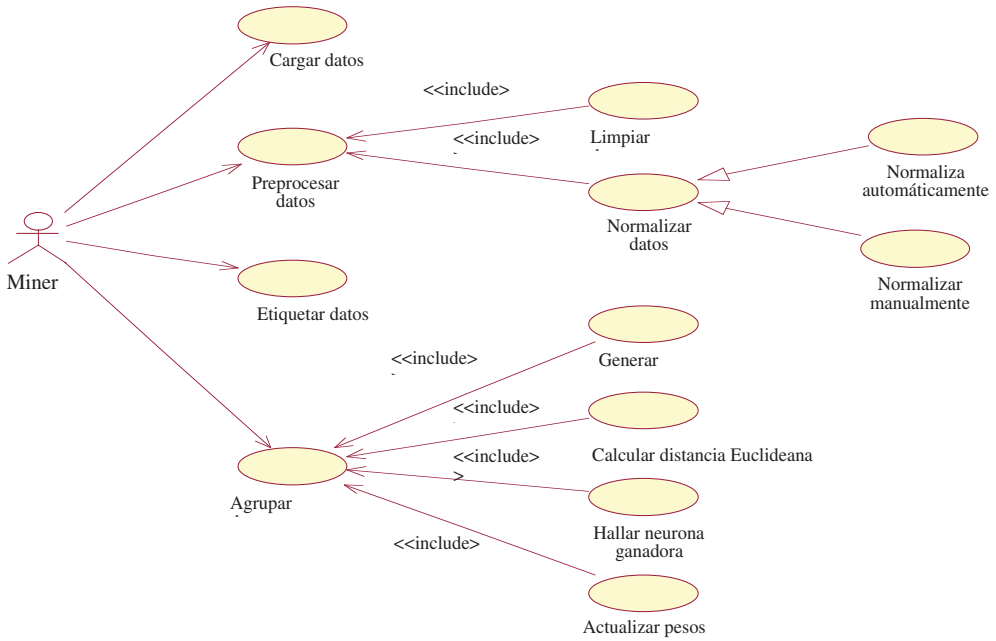
comparación con otros modelos neuronales clasificados dentro de aprendizaje no supervisado.

Para el desarrollo de UDCluster se utilizó Java; este es un lenguaje de programación simple, robusto y portable, de gran aceptación mundial, orientado a objetos e independiente de la plataforma, es decir, del hardware y del sistema operativo. Además, los archivos generados en Java requieren poco espacio para su almacenamiento, convirtiéndolos en altamente portables (Froufe, 2000).

3. Proceso de desarrollo

Para el diseño de la herramienta se utilizó el lenguaje de modelado UML, que permite realizar los diagramas y esquemas de desarrollo de una forma fácil y adecuada. Se presentan a continuación: a) diagrama de casos de uso: refleja el comportamiento del sistema desde el punto de vista del usuario (figura 2); b) diagrama de clases:

Figura 2. UDCluster. Diagrama de casos de uso



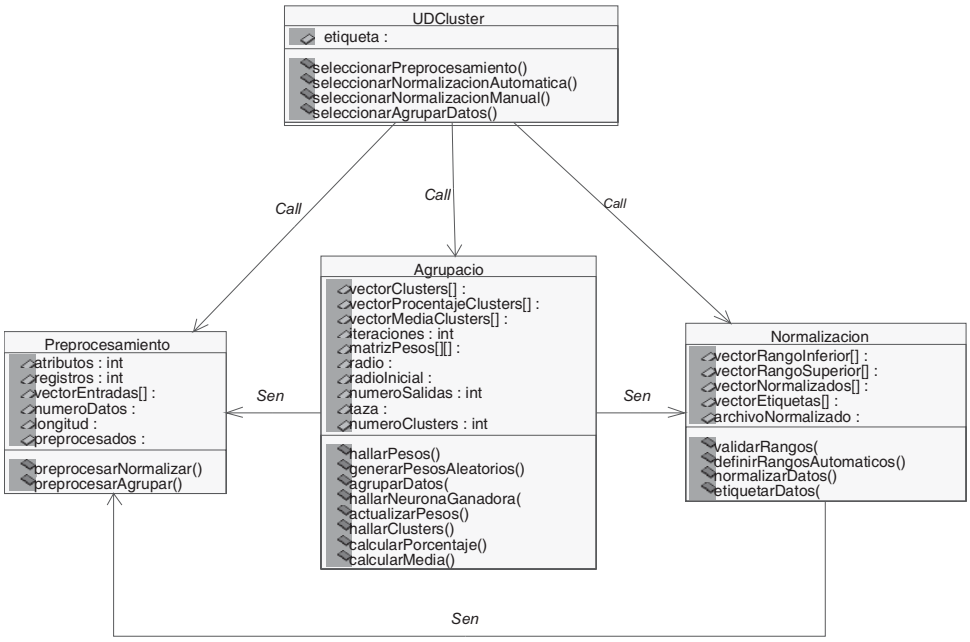
visualiza las clases que componen el sistema y las relaciones entre ellas (Booch, 1999) (figura 3).

En la figura 2 puede observarse que las tareas generales a realizar son:

- Preprocesar datos: los datos del archivo son limpiados y normalizados, para continuar el proceso de agrupación.
- Agrupar datos: los datos del archivo cargado son analizados y agrupados para que puedan ser analizados por el minero. Los pesos de cada conexión entre neuronas de salida y de entrada

que conforman la red neuronal se generan de forma aleatoria (valores entre –2 y 2). El sistema calcula la sumatoria de los datos de entrada y el peso de cada uno, para cada salida generada. Luego se determina la neurona de salida ganadora, esto es, la que generó el menor valor al calcular la distancia euclideana; en otras palabras, aquella que presente mayor similitud entre los datos de entrada y de salida. Por último, en cada iteración se analiza el peso de la neurona ganadora y se actualiza el peso de su vecindario.

Figura 3. Diagrama de clases



En la figura 3 se visualizan las clases que conforman la aplicación desarrollada y las relaciones entre ellas; la interfaz integra numerosas herramientas del paquete Java Swing[3].

UDCluster ejecuta de manera independiente las etapas de preprocesamiento, normalización y agrupación de los datos. El preprocesamiento consiste en la validación y limpieza de los datos almacenados en un archivo (*.txt) que alimenta al sistema; durante este proceso se eliminan espacios en blanco (dos o más seguidos), caracteres no numéricos, exceptuando el punto (.); para datos numéricos reales la coma (,) es reemplazada por punto, y Enter, para diferenciar un patrón de otro.

En el proceso de normalización los datos son estandarizados dentro de un rango especificado, que puede ser -1,0 a 1,0, ó 0,0 a 1,0; se utiliza en particular en algoritmos que involucran redes neuronales, medidas de distancia y agrupación de datos. Por estar todos los datos en un mismo rango, la normalización permite agilizar la fase de aprendizaje en el entrenamiento de redes neuronales.

Uno de los métodos de normalización es el llamado *min-max normalization,* que realiza una transformación lineal de los datos originales; las constantes *minA y maxA* representan los valores mínimo y máximo de un atributo A, y convierten un valor *v* perteneciente a *A* en un dato normalizado *v'*, dentro de los nuevos rangos *new_minA* y *new_maxA*, de acuerdo con la siguiente expresión:

$$v' = \frac{v - minA}{maxA - minA}(new\_maxA - new\_minA) + new\_minA \quad (1)$$

Min-max normalization preserva las relaciones entre los valores de los datos originales (Han, 2001).

A continuación se presenta el algoritmo de entrenamiento:

1) Determinar el vector de entradas, conformado por datos previamente normalizados:

3 Java Swing es una extensión de AWT, que adiciona nuevos componentes y facilita su utilización.

X = [x1, x2,..... xn] (2)

2) Generar pesos. Puede optarse por diferentes formas de generarlos: pesos nulos, iniciación con valores predeterminados (por ejemplo, todos con el mismo valor) o con pequeños valores aleatorios (-2.0 y 2.0), como los utilizados durante esta fase.

W = [wij1, wij2,..... wijn] (3)

3) Calcular la similitud entre el vector de pesos sinápticos *wij* y el vector actual de entrada *x*. Existen diferentes medidas de similitud; por ejemplo, dos vectores serán más similares cuanto menor sea su distancia euclideana. Su función es la siguiente:

$$d^2(w_{ij}, x) = \sum_{k=1}^{n} (w_{ij} - x_k)^2 \quad (4)$$

4) Obtener la neurona ganadora *g,* que será aquella cuyo grado de similitud sea mayor.

5) Calcular la vecindad de cada una de las neuronas de salida con respecto a la ganadora. Para este propósito, se utiliza cualquiera de las funciones de vecindad, dependiendo de la topología deseada (rectangular, triangular, pipa, sombrero mexicano, etc.). También existe la función gaussiana, determinada por:

$$h(|i-g|,t) = \exp\left(\frac{|i-g|^2}{2(Rt)^2}\right) \quad (5)$$

En la ecuación (5) *i* es una de las neuronas de salida, *g* es la neurona ganadora y *R(t)* es el radio dentro del cual se determina la vecindad, determinado por:

$$R(t) = R_0 + (R_f - R_0)\left(\frac{t}{t_R}\right) \quad (6)$$

En (6): R_0 es el radio de vecindad inicial, R_f es el radio de vecindad final, t es la iteración y t_R, el número total de iteraciones.

6) Actualizar pesos. Luego de determinar las neuronas que pertenecen a la vecindad de la ganadora, sus pesos son actualizados por medio de la siguiente expresión:

$$w_{ijk}(t+1) = w_{ijk}(t) + a(t) * h(|i-g|,t) * (x(t) - w_{ijk}(t)) \quad (7)$$

En (7), $a(t)$ es el ritmo de aprendizaje y está definido por:

$$a(t) = a_0 + (a_f - a_o)\left(\frac{t}{t_R}\right) \quad (8)$$

En (8): a_0 es el ritmo de aprendizaje inicial (<1), a_f es el ritmo de aprendizaje final (aproximadamente 0,01), t es la iteración y t_R el número total de iteraciones. El ritmo de aprendizaje y el radio de vecindad disminuyen monótonamente con *t* (número de la iteración actual del proceso de agrupación de datos).

7) Si se ha llegado al número máximo de iteraciones, el proceso de agrupación de los datos finaliza; si no es así, se vuelve al paso 3 y se repite el proceso (Haykin,1999).

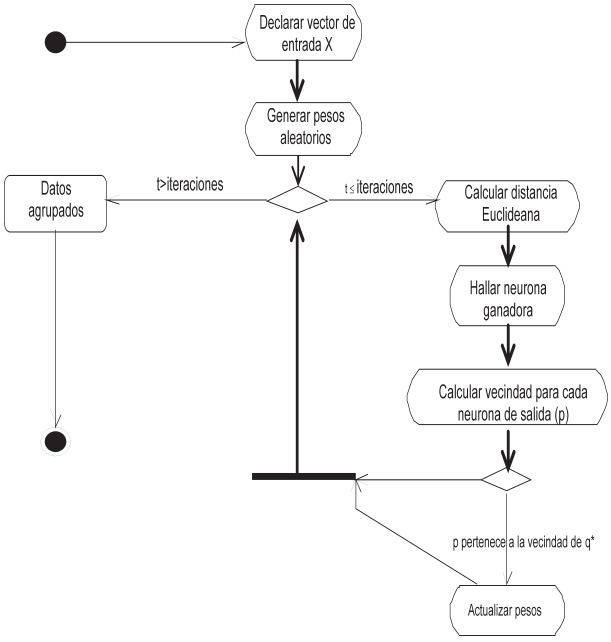


Figura 4. Diagrama de actividad del algoritmo de Kohonen

La figura 4 permite observar las tareas que conforman el algoritmo de Kohonen, y cómo, después de *t* iteraciones, los datos son agrupados.

4. Fases del proceso general de agrupación

Para efectuar el proceso general de agrupación de datos alimentados a la aplicación deben haberse realizado las etapas de preprocesamiento y normalización, las cuales facilitan la manipulación de los datos y garantizan la efectividad de los resultados.

En el preprocesamiento se hace una limpieza de datos y se establece la estructura de representación de la información. UDCluster trabaja con archivos de texto (*.txt), en los cuales los datos son presentados a manera de registros (filas), divididos a su vez en columnas para diferenciar un atributo de otro. En la fase de limpieza se eliminan datos no numéricos, registros y espacios en blanco y se crea un nuevo archivo de extensión (*.udp) con la nueva información. En la normalización, los datos son estandarizados dentro de un rango que puede ser el estipulado por el usuario o, en su defecto, entre 0 y 1.

Una vez normalizados, los datos pueden ser agrupados mediante la aplicación del algoritmo de Kohonen; los resultados son visualizados en un plano (X,Y), que representa todos los datos con respecto a cada uno de los atributos. Además, se presentan los diferentes grupos o *clusters* encontrados, especificando los datos pertenecientes a cada uno y su grado de pertenencia (valor promedio) a un grupo específico. La herramienta también permite conocer el margen de error del proceso de agrupación y, por tanto, el grado de exactitud de los resultados visualizados.

5. ¿Cómo funciona UDCluster?

Para verificar y analizar el funcionamiento de la herramienta se utilizó el archivo de texto llamado *Satimage*, compuesto por una base de datos con imágenes de *Landsat Multi-Spectral Scanner (MSS)*. Una plantilla de MSS consta de cuatro imágenes digitales del mismo paisaje en diferente banda espectral; dos de ellas se encuentran en la región visible (corresponde aproximadamente a regiones verdes y rojas de un espectro visible) y las otras están cerca del infrarrojo. Cada píxel es una palabra binaria de 8 bits: 0 corresponde a negro y 255 a blanco; la resolución espacial de un píxel es aproximadamente de 80*80 m; cada imagen consta de 2.340*3.380 píxeles; por su parte, la base de datos es una subárea diminuta de la escena general y consta de 82*100 píxeles.

Cada línea de datos corresponde a un cuadrado de 3*3 píxeles, completamente contenidos en una subárea de 82*100; contiene los valores de píxel en las cuatro bandas espectrales (convertidas a código ASCII) de cada uno de los nueve píxeles en la vecindad de 3*3, y un número que indica la etiqueta de clasificación del píxel central.

La base de datos contiene 6.435 registros y 36 atributos. Los datos son numéricos, en un rango de 0 a 255 (8 bytes). Hay una etiqueta para cada una de las siguientes clases:

Tabla 1. Etiquetas de clases en la base de datos empleada

Número	*Clase*
1	Red soil
2	Cotton crop
3	Grey soil
4	Damp grey soil
5	Soil with vegetation stubble
6	Mixture class (all types present)
7	Very damp grey soil

En la colección de datos empleada no existen ejemplos con la clase 6; ellos han sido removidos porque existen dudas acerca de su validación. La información correspondiente a cada clase es la siguiente:

Tabla 2. Información contenida según clase

Clase	*Instancias*	*Porcentaje*
1	1.533	23,82 %
2	703	10,92 %
3	1.358	21,10 %
4	626	9,73 %
5	707	10,99 %
7	1.508	23,43 %

El archivo de datos es cargado al sistema para ser validado (figura 5); una vez limpiados los datos, ellos son estandarizados o normalizados para su posterior agrupación (figura 6).

Figura 5. Archivo de datos *Satimage* alimentado al sistema

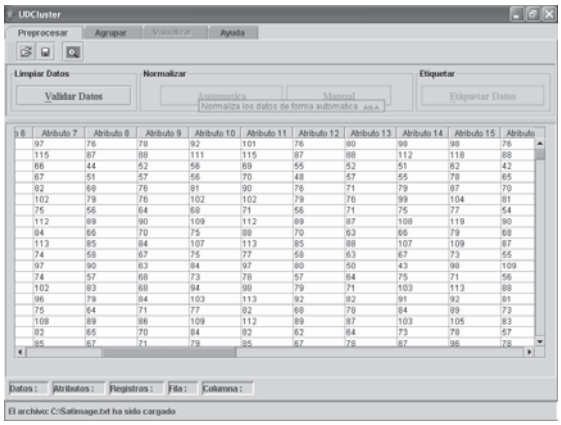
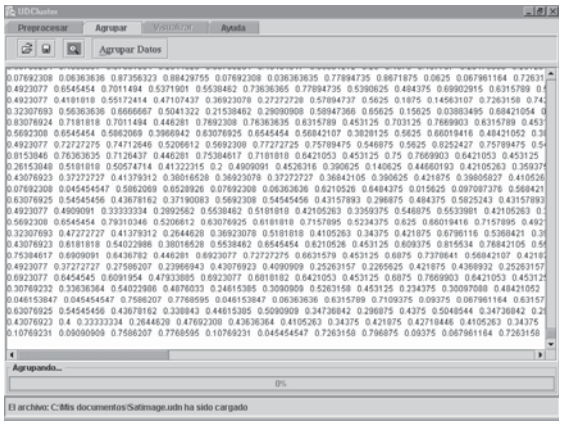
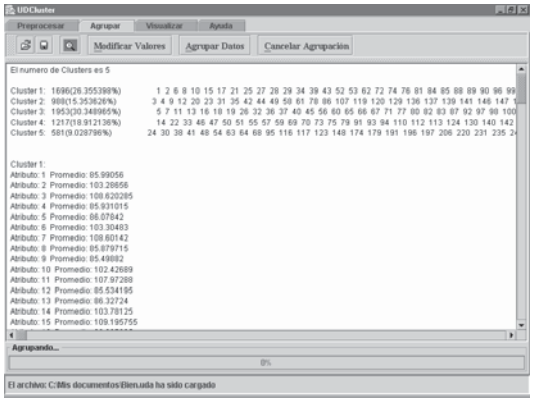


Figura 6. Archivo de datos *Satimage* normalizado



Cuando el archivo de datos *Satimage* es agrupado (figura 7) se generan los siguientes resultados:

Figura 7. Resultados de la agrupación del archivo *Satimage*



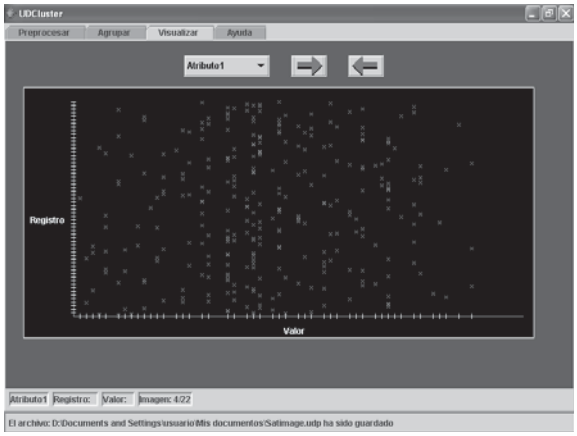
- El número de *clusters* o grupos generados es de cinco.
- El 26,3% de los datos pertenece al grupo 1, el 15,3% pertenece al grupo 2, el 30,3% al grupo 3, el 18,9% al grupo 4 y el 9% al grupo 5.
- Los resultados también visualizan el promedio de los datos de cada atributo de los registros que pertenecen a cada *cluster*. Por ejemplo, en el primer atributo se muestra un promedio de 86, en relación con los datos del primer *cluster* (figura 7). Los resultados pueden ser representados de la siguiente forma:

Tabla 3. Resultados del agrupamiento de datos

Clase	*Instancias*	*Porcentaje*
1	1.696	26,35 %
2	988	15,35 %
3	1.953	30,34 %
4	1.217	18,91 %
5	581	9 %

En la opción de visualización, los datos son graficados de acuerdo con el atributo al cual pertenecen y con base en cada uno de los registros que los conforman (figura 8).

Figura 8. Visualización de los datos del archivo *Satimage*



Los resultados del proceso de agrupación dependen de los parámetros que se especifiquen en la configuración (tasa de aprendizaje, radio de vecindad, número de salidas y número de iteraciones), además de los pesos aleatorios generados al inicio del entrenamiento para establecer las conexiones entre las neuronas de entrada y salida que conforman la red. Por tanto, aunque los resultados obtenidos no representan el mismo número de grupos que se plantean en la matriz del archivo *Satimage*, UDCluster se aproxima bastante a ellos y da la posibilidad de mejorarlos y disminuir el margen de error, modificando nuevamente los parámetros de configuración.

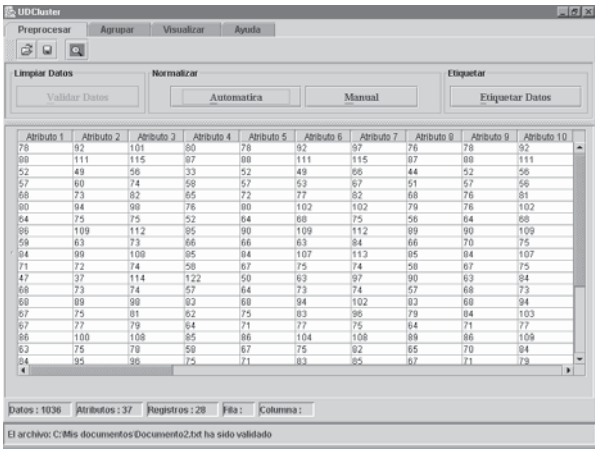
5.1. Procesos efectuados para la agrupación de datos

UDCluster permite la agrupación de datos previamente almacenados en un archivo plano y luego importados a la aplicación. Las principales funciones de esta herramienta son: a) cargar archivo; b) preprocesar datos; c) visualizar datos preprocesados; d) normalizar datos; e) visualizar datos normalizados; f) etiquetar datos; g) agrupar datos; h) graficar datos; i) guardar archivos generados.

Para la agrupación de los datos, la aplicación cuenta con una barra de herramientas en la parte superior y en el centro una ficha con cuatro páginas que especifican cada una de las fases de este proceso.

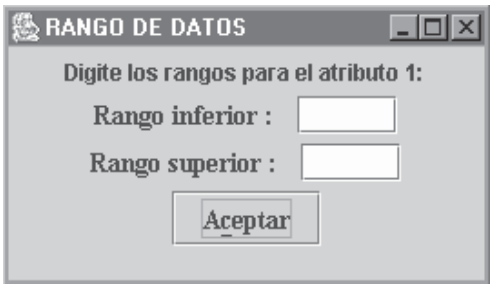
Una vez iniciada la herramienta, el archivo es cargado en un área de texto para su validación; en ella cada dato es analizado y validado para su utilización durante los procesos restantes; los nuevos datos se almacenan en un nuevo archivo con extensión *.udp.

Figura 9. Visualización de datos preprocesados



En la figura 9 se observa que la herramienta muestra los datos en una tabla después de la fase de validación, informando al usuario el número de datos, atributos y registros que conforman el archivo cargado, una vez éste ha sido limpiado. La segunda fase es la normalización del archivo .udp; en ella los datos son estandarizados dentro de un mismo rango; el usuario puede llevar a cabo este proceso de forma manual o automática. En la normalización manual, los rangos (superior e inferior) deben ser digitados para cada atributo (fila) existente en el archivo; en la normalización automática, estos se generan a partir del menor y mayor valor de los datos encontrados en cada atributo. Al igual que en la validación, se genera un archivo con extensión *.udn. En la figura 10 se observa el *frame* que emerge cuando el usuario opta por la normalización manual de datos.

Figura 10. Normalización manual



Un proceso adicional es etiquetar datos, que consiste en nombrar cada atributo o columna del archivo cargado, facilitando su interpretación; por defecto, la herramienta asigna un nombre específico a los atributos (la palabra Atributo*,* más el número al que corresponda: Atributo 1, Atributo 2).

En la tercera fase se lleva a cabo el entrenamiento de la red para la agrupación de los datos; a medida que avanza el proceso, una barra de progreso mostrará al usuario el ritmo de entrenamiento. Cuando finaliza la agrupación, se muestra el número de *clusters* generados, el número de registros que contiene cada *cluster* con su respectivo porcentaje y el promedio de cada atributo en cada uno de los grupos.

La figura 11 muestra los resultados de la agrupación de los datos. Además de las variables de salida arriba mencionadas, se observa el error cuantificador promedio, que determina el grado de optimización de los resultados; el usuario puede guardar el archivo generado, que tiene una extensión *.uda. Si desea modificar los valores utilizados para entrenar la red y optimizar los resultados debe hacer clic sobre el botón Modificar Valores, desplegando la ventana de ingreso de datos (figura 12).

Figura 11. Resultados del entrenamiento de la red

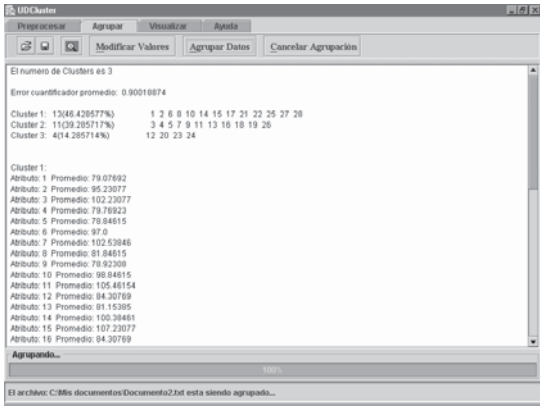
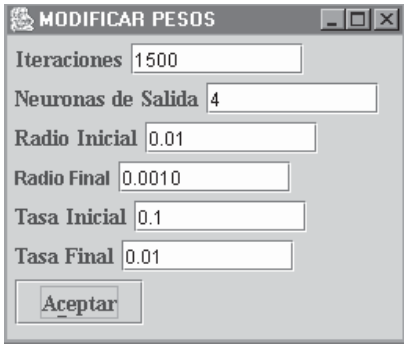
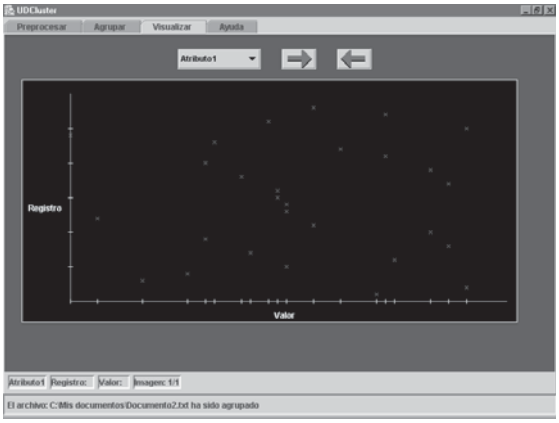


Figura 12. Ventana para modificar valores de entrenamiento



Una vez normalizados los datos, estos son visualizados en una gráfica. El usuario debe escoger de un menú el atributo que quiere graficar. En la figura 13 se observa el valor del atributo 2 (sin normalizar) en un plano de coordenadas donde *Y* es el número del registro y el *X* es el atributo escogido por el usuario a través del menú.

Figura 13. Visualización de los datos



6. Conclusiones

Luego de aplicar minería de datos al archivo *Satimage,* los resultados obtenidos por UDCluster

arrojan el mismo número de grupos (*clusters*) que se obtienen con otras herramientas de similares características, por ejemplo WEKA[4]. Se concluye que los resultados obtenidos son satisfactorios, alcanzándose un primer resultado en el marco del proyecto de investigación UDMiner.

El desarrollo permite observar que el modelo neuronal de Kohonen es ideal para encontrar relaciones entre grupos complejos, al basarse en el manejo de vecindad para agrupar; sin embargo, se identifica como limitante el largo tiempo computacional que consume para construir los grupos.

En relación con otras herramientas de propósito similar, como WEKA o CBA, UDCluster pretende mejorar la interacción con el usuario, mediante una interfaz más cómoda a la hora de realizar todo el proceso de agrupación, orientando de alguna forma al usuario a través de los distintos procesos mediante el uso de pestañas. Además, la complejidad de la herramienta disminuye si se evalúa la estructura de los archivos, que pueden ser cargados desde un inicio para su posterior agrupamiento; pudo observarse que UDCluster requiere solo de los datos que componen el archivo, y de ninguna otra especificación para realizar el proceso; esta característica lo diferencia de WEKA.

No obstante, cabe también anotar que el prototipo de *software* aquí presentado debe ser sometido a actualizaciones para la obtención de mejores resultados en cuanto a generación de grupos, para reducir la complejidad de los algoritmos de agrupación y preprocesamiento empleados, de tal forma que se optimice el tiempo de procesamiento.

La herramienta hace parte del proyecto conjunto UDMiner (Herramientas de desarrollo para la minería de datos), que pretende comparar la efectividad del algoritmo Kohonen, con respecto a otros modelos. Mediante trabajos futuros se espera desarrollar una herramienta de similares características a UDCluster, utilizando el modelo neuronal artificial de resonancia adaptativa ART.

Como aporte del trabajo se destaca el desarrollo de este tipo de herramientas en Colombia, pues a la fecha solo se conoce de la existencia de dos desarrollos en la Universidad Nacional de Colombia. Diferentes empresas colombianas han adoptado esta tecnología, pero todo se limita a la compra de software a entidades extrajeras.

Referencias bibliográficas

[1] BERRY, M. (1997). *Data mining techniques for marketing*. New York, Kaufmann Publishers.

[2] BOOCH, G; I. JACOBSON y J. RUMBAU-GH (1999). *Lenguaje unificado de modelado*. Madrid, Addison Wesley.

[3] FROUFE, A. (2000). *Java 2. Manual de usuario y tutorial*. México D.F., Alfaomega Ra-Ma.

[4] HAN, J. y M. KAMBER (2001). *Data Mining. Concepts and Techniques*. San Francisco, Kauffman Morgan.

[5] HAYKIN, S. (1999). *Neural Networks, a comprehensive foundation*, 2 ed. New York, Prentice Hall.

[6] HILERA, J. y V. MARTÕNEZ (1996). *Redes neuronales artificiales. Fundamentos, modelos y aplicaciones*. Madrid, Alfaomega Ra-Ma.

4 WEKA es una herramienta informática utilizada para minería de datos, desarrollada en la Universidad de Waikato - Nueva Zelanda.