Sistemas difusos para computación en Big Data
Ecuaciones no resolubles con respecto a la derivada

Doble Grado en Ingeniería Informática y Matemáticas

Antonio Coín Castro

17 de Septiembre de 2020

Trabajo Fin de Grado

E.T.S de Ingenierías Informática y de Telecomunicación
Facultad de Ciencias

UNIVERSIDAD DE GRANADA

Índice de contenidos

Sistemas difusos para computación en Big Data

Introducción

El problema de aprendizaje de datos es un tema central en el aprendizaje automático. Una propuesta relevante en este sentido son los sistemas basados en reglas difusas, que permiten resolver problemas de forma aproximada pero efectiva.

Por otro lado, en la *era de la información* las cantidades de datos que se manejan son cada vez mayores, y surge el concepto de Big Data. ¿Están preparados los algoritmos existentes para tratar grandes cantidades de datos?

Solución: construir sistemas difusos **escalables**.

Objetivos

- Estudiar la teoría de conjuntos difusos, la lógica difusa y los sistemas difusos desde un punto de vista teórico.

Objetivos

- Estudiar la teoría de conjuntos difusos, la lógica difusa y los sistemas difusos desde un punto de vista teórico.
- Definir el concepto de Big Data, sus características y la infraestructura asociada. Estudiar el modelo MapReduce.

Objetivos

- Estudiar la teoría de conjuntos difusos, la lógica difusa y los sistemas difusos desde un punto de vista teórico.
- Definir el concepto de Big Data, sus características y la infraestructura asociada. Estudiar el modelo MapReduce.
- Diseñar, implementar y probar una serie de sistemas difusos para computación escalable.

Sistemas difusos para computación en Big Data

Conjuntos y lógica difusa

Principio de incompatibilidad

"As the complexity of a system increases, our ability to make precise and yet significant statements about its behavior diminishes until a threshold is reached beyond which precision and significance become almost mutually exclusive characteristics."

Lofti A. Zadeh, Fuzzy Sets.

Función de pertenencia

Si X es un universo de objetos y $A \subseteq X$ un conjunto, existe implícitamente una **función de pertenencia**:

$$\mu_A(x) = \begin{cases} 1, & \text{si } x \in A \\ 0, & \text{si } x \notin A. \end{cases}$$

Función de pertenencia

Si X es un universo de objetos y $A \subseteq X$ un conjunto, existe implícitamente una **función de pertenencia**:

$$\mu_A(x) = \begin{cases} 1, & \text{si } x \in A \\ 0, & \text{si } x \notin A. \end{cases}$$

Podemos permitir que la función de pertenencia tome un continuo de valores posibles:

$$\mu_A : X \longrightarrow [0,1], \quad x \mapsto \mu_A(x).$$

Esta función le asigna a cada elemento x del universo un **grado de pertenencia** al conjunto A.

Definición (Conjunto difuso)

Un conjunto difuso A en X es el conjunto de pares ordenados

$$A = \{(x, \mu_A(x)) \mid x \in X\}$$

El conjunto A queda determinado por la función μ_A.

Conjuntos difusos

Definición (Conjunto difuso)

Un conjunto difuso A en X es el conjunto de pares ordenados

$$A = \{(x, \mu_A(x)) \mid x \in X\}$$

El conjunto A queda determinado por la función μ_A.

Permiten modelar la imprecisión y la ambigüedad. Se permite el solapamiento.

Ejemplo

$A =$ *"una persona jóven"*, $B =$ *"sobre 30 años de edad"*.

Teoría de conjuntos difusos

- Definiciones y resultados básicos.
- Operaciones difusas: unión, intersección, negación, producto Cartesiano...
- Relaciones difusas.
- Funciones de pertenencia usuales.
- Operadores difusos: T-normas y T-conormas.

Lógica difusa

Lógica multivaluada con tantos valores de verdad como números reales hay en $[0,1]$. Se apoya en el concepto de variables lingüísticas.

Ejemplo

T(*edad*) = { *joven, no joven, muy joven, no muy joven, de mediana edad, viejo, no viejo, muy viejo, más o menos viejo, no muy viejo, no muy joven y no muy viejo, . . .* }.

Cada término en $T(edad)$ viene caracterizado por un conjunto difuso en $X = [0,100]$.

Reglas difusas de tipo Si-Entonces

Consideramos A y B valores lingüísticos en dos universos X e Y, respectivamente. Estudiaremos reglas del tipo

Si x es A entonces y es B.

"x es A" $\rightarrow$ **antecedente**.
"y es B" $\rightarrow$ **consecuente**.

Podemos ver esta implicación como un conjunto difuso $\mathcal{R} \equiv A \rightarrow B \equiv f(\mu_A(x), \mu_B(y))$.

Razonamiento difuso

El razonamiento difuso se puede entender como un *modus ponens* generalizado.

$$\frac{\begin{array}{c} x \text{ es } A' \\ \text{si } x \text{ es } A \text{ entonces } y \text{ es } B \end{array}}{y \text{ es } B'}$$

Entonces $B' = A' \circ (A \rightarrow B)$, donde $\circ$ es un operador de composición.

Podemos evaluar la presencia de varios antecedentes con el operador de producto cartesiano, y la de varias reglas con el operador de unión.

Sistemas de inferencia difusos

Un sistema de inferencia difuso recibe una entrada y produce una respuesta utilizando razonamiento difuso. Consta de cuatro componentes.

- Un **módulo de fuzzificación** con funciones de pertenencia para transformar la entrada en conjuntos difusos.
- Una **base de reglas** que contiene un conjunto de reglas difusas de tipo si-entonces.
- Un **mecanismo de razonamiento**, que realiza el procedimiento de inferencia.
- Un **mecanismo de defuzzificación** para producir una respuesta nítida (opcional).

Tipos de sistemas difusos

Mamdani. Emplea reglas del tipo

$$\text{si } X_1 \text{ es } A_1 \text{ y } \dots \text{ y } X_n \text{ es } A_n$$
$$\text{entonces } Y_1 \text{ es } B_1 \text{ y } \dots \text{ y } Y_m \text{ es } B_m.$$

Tipos de sistemas difusos

Mamdani. Emplea reglas del tipo

$$\text{si } X_1 \text{ es } A_1 \text{ y } \dots \text{ y } X_n \text{ es } A_n$$
$$\text{entonces } Y_1 \text{ es } B_1 \text{ y } \dots \text{ y } Y_m \text{ es } B_m.$$

TSK. Define reglas de la forma

$$\text{si } X_1 \text{ es } A_1 \text{ y } \dots \text{ y } X_n \text{ es } A_n$$
$$\text{entonces } Y_1 \text{ es } f_1(X_1, \dots, X_n) \text{ y } \dots \text{ y } Y_m \text{ es } f_m(X_1, \dots, X_n),$$

donde cada f_i es una función nítida de la entrada.

Sistema difuso de tipo Mamdani

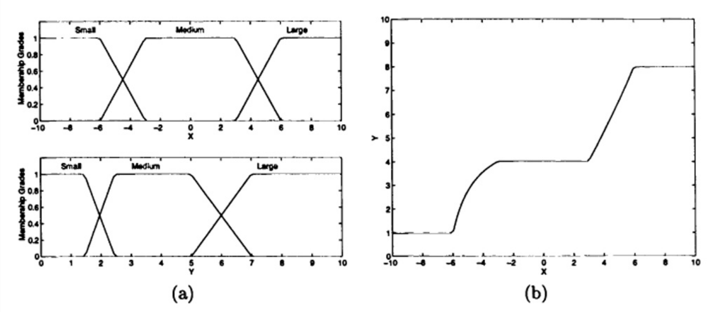

Figura 1: Representación es un sistema de Mamdani de (*a*) funciones de pertenencia de antecedentes y consecuentes; y (*b*) curva de entrada-salida tras defuzzificar.

Sistema difuso de tipo TSK

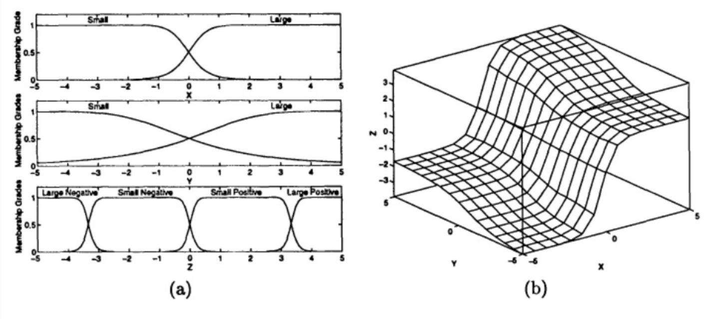

Figura 2: Representación en un sistema TSK de (a) funciones de pertenencia de antecedentes y consecuentes; y (b) curva de entrada-salida.

Sistemas difusos para computación en Big Data

Fundamentos de Big Data

Las cinco V's

Utilizamos los datos para **resolver un problema**.

- Volumen
- Velocidad
- Variedad
- Veracidad
- Valor

Modelo de programación distribuida propuesto por Google en 2004.

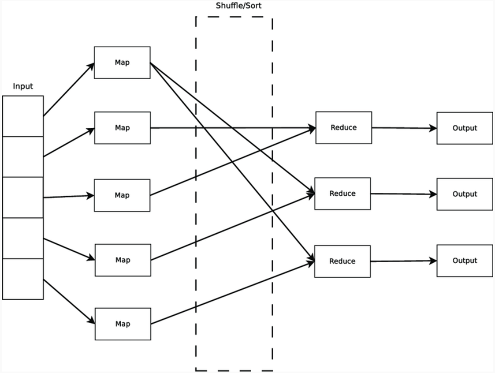

Figura 3: Esquema de la arquitectura MapReduce.

Sistemas difusos para computación en Big Data

Diseño e implementación de algoritmos

Algoritmos difusos de aprendizaje

Se clasifican principalmente en tres tipos:

- Basados en **particiones**. Crean una partición difusa del espacio para medir cómo encajan en ellas los datos y construir reglas en consecuencia.
- **Neurodifusos**. A partir de una estructura inicial de reglas se emplean modelos neuronales para *ajustar* los parámetros de las funciones de pertenencia.
- **Genéticos**. Similares a los anteriores, pero se emplean algoritmos genéticos para la fase del ajuste.

Algoritmo de Wang y Mendel

Propuesto por Wang y Mendel en 1992 para construcción de reglas a partir de datos.

1. Dividir el espacio de entrada en segmentos difusos.

Algoritmo de Wang y Mendel

Propuesto por Wang y Mendel en 1992 para construcción de reglas a partir de datos.

1. Dividir el espacio de entrada en segmentos difusos.
2. Calcular la pertenencia de cada punto a todas las regiones, y quedarnos en cada caso con las de máxima pertenencia. Así formamos una regla difusa.

Algoritmo de Wang y Mendel

Propuesto por Wang y Mendel en 1992 para construcción de reglas a partir de datos.

1. Dividir el espacio de entrada en segmentos difusos.
2. Calcular la pertenencia de cada punto a todas las regiones, y quedarnos en cada caso con las de máxima pertenencia. Así formamos una regla difusa.
3. Calculamos un peso o *importancia* para cada regla: el producto de la pertenencia del punto a todas las regiones.

Algoritmo de Wang y Mendel

Propuesto por Wang y Mendel en 1992 para construcción de reglas a partir de datos.

1. Dividir el espacio de entrada en segmentos difusos.
2. Calcular la pertenencia de cada punto a todas las regiones, y quedarnos en cada caso con las de máxima pertenencia. Así formamos una regla difusa.
3. Calculamos un peso o *importancia* para cada regla: el producto de la pertenencia del punto a todas las regiones.
4. Simplificamos la base de datos eliminando duplicados y conflictos eligiendo siempre las reglas de mayor importancia.

Algoritmo de Wang y Mendel

Propuesto por Wang y Mendel en 1992 para construcción de reglas a partir de datos.

1. Dividir el espacio de entrada en segmentos difusos.
2. Calcular la pertenencia de cada punto a todas las regiones, y quedarnos en cada caso con las de máxima pertenencia. Así formamos una regla difusa.
3. Calculamos un peso o *importancia* para cada regla: el producto de la pertenencia del punto a todas las regiones.
4. Simplificamos la base de datos eliminando duplicados y conflictos eligiendo siempre las reglas de mayor importancia.
5. Para la predicción, utilizamos el método de defuzzificación COA.

Implementación del algoritmo WM

Etapa map. Para cada punto calculamos la regla asociada y su importancia.

Etapa reduce. Agregamos todas las reglas, eliminando conflictos y duplicados.

```scala
val ruleBase = data.mapPartitions { case (x, y) =>
  val (ri, ro, degreeIn, degreeOut)
    = maxRegion(x, y, regionsIn, regionsOut)

  (ri, (ro, degreeIn * degreeOut))
}.reduceByKey { case (r1, r2) =>
  if (r1._2 > r2._2) r1 else r2
}.map { case (ri, (ro, _)) => (ri, ro) }
```

Algoritmo subtractive clustering

Propuesto por S. Chiu en 1994 para encontrar número y valor incial de centroides en clústering difuso.

Idea: cada punto es un posible centroide, y buscamos centroides suficientemente separados. Se asigna inicialmente a cada punto un potencial inversamente proporcional a la distancia a todos los demás.

Algoritmo subtractive clustering

Propuesto por S. Chiu en 1994 para encontrar número y valor incial de centroides en clústering difuso.

Idea: cada punto es un posible centroide, y buscamos centroides suficientemente separados. Se asigna inicialmente a cada punto un potencial inversamente proporcional a la distancia a todos los demás.

1. Se inicializa el potencial de cada punto y se elige aquel con mayor potencial como centroide.

Algoritmo subtractive clustering

Propuesto por S. Chiu en 1994 para encontrar número y valor incial de centroides en clústering difuso.

Idea: cada punto es un posible centroide, y buscamos centroides suficientemente separados. Se asigna inicialmente a cada punto un potencial inversamente proporcional a la distancia a todos los demás.

1. Se inicializa el potencial de cada punto y se elige aquel con mayor potencial como centroide.
2. Se recalculan los potenciales, que disminuyen de forma proporcional a la distancia al centroide elegido.

Algoritmo subtractive clustering

Propuesto por S. Chiu en 1994 para encontrar número y valor incial de centroides en clústering difuso.

Idea: cada punto es un posible centroide, y buscamos centroides suficientemente separados. Se asigna inicialmente a cada punto un potencial inversamente proporcional a la distancia a todos los demás.

1. Se inicializa el potencial de cada punto y se elige aquel con mayor potencial como centroide.
2. Se recalculan los potenciales, que disminuyen de forma proporcional a la distancia al centroide elegido.
3. Se repite el proceso hasta que el potencial cae por debajo de un umbral.

Versiones del algoritmo subtractive clustering

1. Verisón **global**. Para inicializar el potencial se consideran todos los puntos, creando todas las posibles parejas.
2. Versión **local**. Se distribuyen los puntos en particiones y el algoritmo se aplica localmente. Después se concatenan los resultados.
3. Versión **híbrida**. Se aplica la versión local, se crean grupos de particiones y en ellos se aplica la versión global, utilizando los centroides como datos de entrada.

Versiones del algoritmo subtractive clustering

1. Verisón **global**. Para inicializar el potencial se consideran todos los puntos, creando todas las posibles parejas.
2. Versión **local**. Se distribuyen los puntos en particiones y el algoritmo se aplica localmente. Después se concatenan los resultados.
3. Versión **híbrida**. Se aplica la versión local, se crean grupos de particiones y en ellos se aplica la versión global, utilizando los centroides como datos de entrada.

Extensión a regresión: cada centroide es una regla difusa. Utilizamos defuzzificación COA.

Extensión a clasificación: asignar a cada punto la etiqueta del clúster con mayor pertenencia.

Algoritmo Fuzzy C-Means

Propuesto por J. Dunn en 1973 y mejorado por J. Bezdek en 1981, para realizar clústering difuso.

El i-ésimo punto tendrá una pertenencia al j-ésimo clúster, digamos u_{ij}. Se pretende optimizar la función

$$J_m = \sum_X \sum_V u_{ij}^m \|x_i - v_j\|^2,$$

sujeta a las restricciones

$$u_{ij} > 0 \ \ \forall i, j, \quad y \quad \sum_V u_{ij} = 1 \ \ \forall i.$$

Se resuelve utilizando los multiplicadores de Lagrange para obtener unas reglas iterativas de actualización de la matriz de pertenencias y los centroides.

Implementación del algoritmo FCM

$$u_{ij} = \frac{1}{\displaystyle\sum_{k=1}^{c} \left(\frac{\|x_i - v_j\|}{\|x_i - v_k\|} \right)^{\frac{2}{m-1}}}, \qquad v_j = \frac{\displaystyle\sum_{i=1}^{n} u_{ij}^m x_i}{\displaystyle\sum_{i=1}^{n} u_{ij}^m}.$$

Etapa map. Se calculan para cada punto los valores necesarios para actualizar los centroides. No se mantiene en memoria la matriz de pertenencia.

Etapa reduce. Para cada centroide, se suman entre sí los valores obtenidos en la etapa anterior.

Extensión a clasificación: procedemos como en el algoritmo anterior. En este caso se asigna a cada clúster la etiqueta de la clase mayoritaria tras un conveniente α-corte.

Clústering duro vs clústering difuso

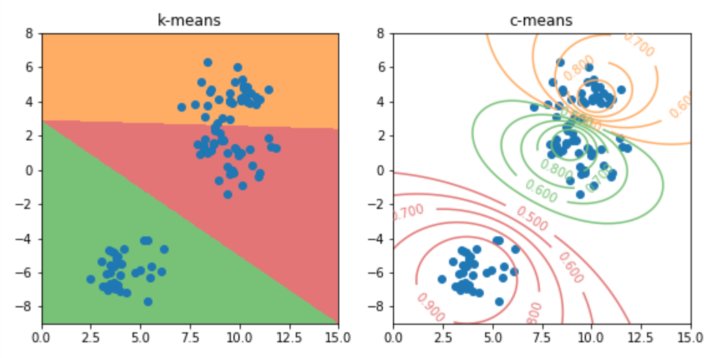

Figura 4: Comparación del algoritmo de clústering clásico K-means y el algoritmo de clústering difuso Fuzzy C-Means.

Sistemas difusos para computación en Big Data

Estudio comparativo

Comparación de precisión en clasificación

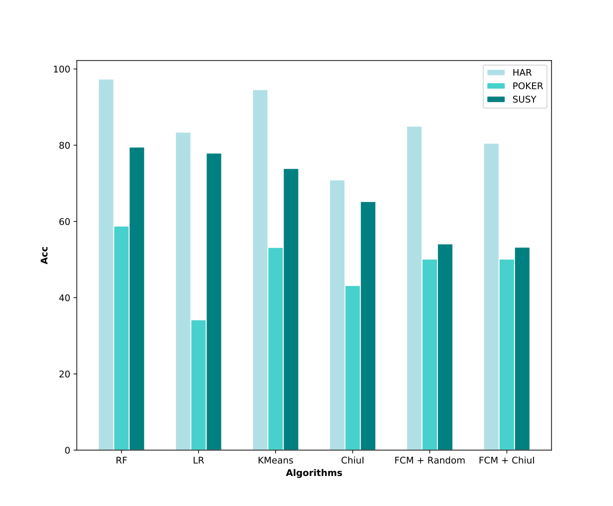

Figura 5: Comparación de la precisión obtenida en los tres conjuntos de datos elegidos para los algoritmos de clasificación.

Comparación de tiempo en clasificación

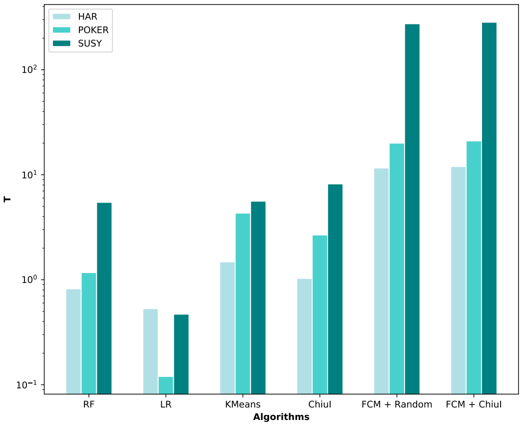

Figura 6: Comparación del tiempo de ejecución en los tres conjuntos de datos elegidos para los algoritmos de clasificación.

Comparación de error en regresión

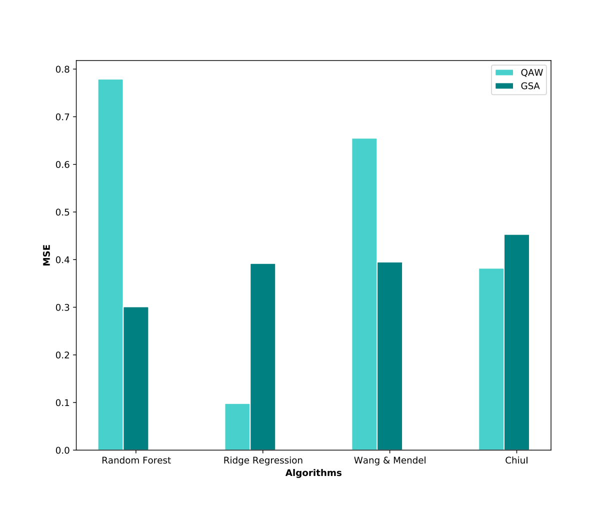

Figura 7: Comparación del Mean Squared Error en los dos conjuntos de datos elegidos para los algoritmos de regresión.

Comparación de tiempo en regresión

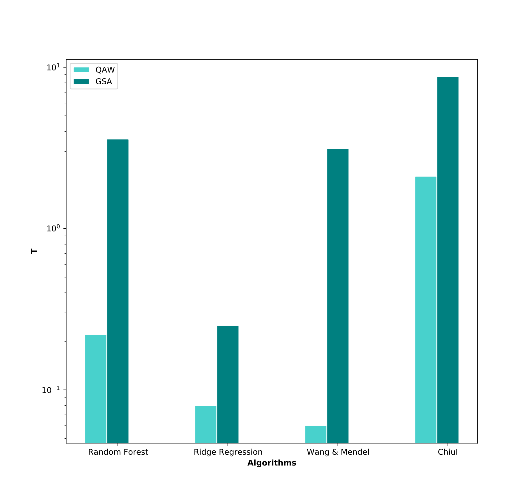

Figura 8: Comparación del tiempo de ejecución en los dos conjuntos de datos elegidos para los algoritmos de regresión.

Conclusiones

- Hemos realizado una aportación original desarrollando algoritmos escalables para aprendizaje de sistemas difusos, justificando su interés teórico y práctico.

Conclusiones

- Hemos realizado una aportación original desarrollando algoritmos escalables para aprendizaje de sistemas difusos, justificando su interés teórico y práctico.
- Hemos comprobado que es necesario un diseño cuidadoso para adaptar los algoritmos al escenario Big Data. No basta una mera paralelización.

Conclusiones

- Hemos realizado una aportación original desarrollando algoritmos escalables para aprendizaje de sistemas difusos, justificando su interés teórico y práctico.
- Hemos comprobado que es necesario un diseño cuidadoso para adaptar los algoritmos al escenario Big Data. No basta una mera paralelización.
- Nos hemos familiarizado con el modelo de referencia MapReduce y las herramientas del estado del arte en el Big Data, como Apache Spark y HDFS.

Referencias

- JANG, J. S. R., SUN, C. T., & MIZUTANI, E. *Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence.* (1997).
- L.X. WANG, J. M. MENDEL. *Generating fuzzy rules by learning from examples.* (1992).
- S. CHIU. *Fuzzy Model Identification Based on Cluster Estimation.* (1994).
- JAMES C. BEZDEK *ET AL.* *Detection and Characterization of Cluster Substructure I. Linear Structure: Fuzzy c-Lines.* (1981).

Ecuaciones no resolubles con respecto a la derivada

Gracias por su atención