APRENDIZAJE HÍBRIDO DE LA BASE DE CONOCIMIENTO DE UN SISTEMA BASADO EN REGLAS DIFUSAS MEDIANTE ALGORITMOS GENÉTICOS Y COLONIAS DE HORMIGAS

J. Casillas, O. Cordón, F. Herrera
Dept. Ciencias de la computación e inteligencia artificial
E.T.S. Ing. Informática
Universidad de Granada, 18071 - Granada
casillas,ocordon,herrera@decsai.ugr.es

P. Villar
Dept. Informática
E.S. de Ing. Informática
Universidade de Vigo, 32004 - Ourense
pvillar@uvigo.es

Resumen

En muchos ejemplos de modelado difuso lingüístico se hace necesario un aprendizaje automático de la base de conocimiento a partir de datos numéricos. En este trabajo, se propone un método para aprender la totalidad de la base de conocimiento (particiones difusas y reglas difusas), que combina un algoritmo genético y un algoritmo de optimización mediante colonias de hormigas. Estos dos mecanismos trabajan de forma coordinada para conseguir modelos con buen equilibrio entre precisión y interpretabilidad.

Palabras Clave: Aprendizaje automático, sistemas basados en reglas difusas, algoritmos genéticos, optimización mediante colonias de hormigas.

1 INTRODUCCIÓN

Actualmente, el modelado difuso lingüístico es una aplicación importante de los Sistemas Basados en Reglas Difusas (SBRDs). En este tipo de modelos, la interpretabilidad es un requisito fundamental, junto con la precisión. Para esta tarea, se utilizan los SBRDs Lingüísticos (SBRDLs), donde las reglas difusas toman valores de un conjunto de términos con una interpretación en el mundo real (variables lingüísticas).

La Base de Conocimiento (BC) de un SBRDL está formada por dos componentes: 1) la Base de Reglas (BR), constituida por el conjunto de reglas difusas, y 2) la Base de Datos (BD), que incluye las funciones de pertenencia de cada uno de los términos lingüísticos (o etiquetas) de las particiones difusas asociadas a las variables. La BC de un SBRDL depende del problema que se intente resolver, por tanto, se hace necesario algún mecanismo de derivación automática de dicha BC. Dentro de la literatura especializada existen numerosos métodos de aprendizaje de la BR a partir de una BD previamente definida.En la mayoría de esos métodos, se consideran particiones difusas uniformes con el mismo número de etiquetas para todas las variables. Sin embargo, la elección de la BD tiene una decisiva influencia en el comportamiento del SBRDL resultante[5].

Teniendo en cuenta la fuerte interrelación entre los dos componentes de la BC, sería muy aconsejable un mayor grado de cooperación entre las tareas de aprendizaje de la BR y la BD, que permitiese obtener modelos con un buen equilibrio entre interpretabilidad y precisión. Basándonos en esta idea, se propone un método de aprendizaje de la BC que mantiene una población de posibles definiciones de la BD al mismo tiempo que busca un conjunto compacto de reglas (BR). Más concretamente, para la derivación de las reglas difusas utilizaremos un mecanismo basado en la metodología COR junto con un algoritmo de optimización mediante colonias de hormigas (proceso COR-OCH). Para la definición de la BD se empleará un Algoritmo Genético que incluye sus principales componentes para cada variable: dominio, número de etiquetas y partición difusa. En este documento, describiremos primero el mecanismo de aprendizaje COR-OCH. A continuación, se presenta el método propuesto y finalmente, se muestran unos resultados experimentales y algunas conclusiones.

2 PROCESO DE APRENDIZAJE COR-OCH

El proceso seguido para integrar el aprendizaje de la base de reglas dentro de un meta-algoritmo de diseño de la base de datos debe ser rápido y eficaz. Para ello, basaremos el aprendizaje en la metodología COR (Cooperación entre Reglas) propuesta en [1] y extendida

[0]Trabajo parcialmente soportado por el Ministerio de Ciencia y Tecnología en el marco de los proyectos TIC2002-04036-C05-01, TIC2003-00877 (con fondos FEDER) y de la Red HEUR TIC2002-10866-E.

en [2] usando un algoritmo de optimización mediante colonias de hormigas (OCH) [7]. La reducción del espacio de búsqueda e inducción de cooperación entre reglas realizada por la metodología COR, junto con el uso de información heurística y de una búsqueda eficaz de la técnica OCH, permiten que el algoritmo propuesto sea rápido y genere bases de reglas con buena precisión y alta interpretabilidad. Las siguientes dos subsecciones describen la metodología de aprendizaje COR y el algoritmo de OCH basado en ella.

2.1 Metodología COR

En los últimos años se ha propuesto en la literatura especializada una familia de métodos simples y eficientes de extracción de reglas difusas guiados por criterios de cobertura de los datos del conjunto de ejemplos. Su simplicidad, además de su velocidad y fácil comprensión, hace a estos métodos muy adecuados para tareas de aprendizaje. Sin embargo, estos métodos generalmente buscan reglas difusas con el mejor comportamiento individual (por ejemplo, [11]) y, por tanto, no se considera la interacción global entre las reglas debida al razonamiento interpolado realizado por los sistemas difusos, provocando así una mala precisión en algunos casos.

Con la intención de paliar estos inconvenientes manteniendo las interesantes ventajas de estos métodos, en [1] se propone la metodología COR. En lugar de seleccionar el consecuente con el mejor comportamiento en cada subespacio, esta metodología considera la posibilidad de usar otro consecuente, diferente del mejor, si así se mejora el comportamiento global del sistema difuso gracias a tener una base de reglas con mejor cooperación.

COR se compone de los siguientes pasos: (1) *construcción del espacio de búsqueda*, se obtiene un conjunto de consecuentes candidatos para cada regla; y (2) *selección del conjunto de reglas más cooperativo*, se realiza una búsqueda combinatoria entre los conjuntos para encontrar la combinación de consecuentes con el mejor comportamiento global. En la figura 1 se muestra una descripción más detallada del proceso de generación de reglas basado en COR.

2.2 Algoritmo de OCH basado en COR

Dado que el espacio de búsqueda manejado en el paso 2. es grande, resulta necesario usar técnicas de búsqueda aproximadas. En [1] se obtuvieron modelos lingüísticos con buena precisión utilizando enfriamiento simulado. Sin embargo, dado que una de las restricciones de nuestro problema es que el proceso de aprendizaje debe ser rápido, en este artículo se propone el uso de OCH [7]. Esta es una técnica de búsqueda bio-inspirada que considera información heurística para permitir obtener buenas soluciones con rapidez. Esta sección describe los componentes del algoritmo propuesto.

Representación del problema Para aplicar OCH en la metodología COR es conveniente verla como un problema de optimización combinatoria con la capacidad de ser representado en un grafo ponderado. De esta forma, se puede afrontar el problema considerando un número fijo de subespacios e interpretando el proceso de aprendizaje como la forma de asignar vectores de consecuentes —es decir, etiquetas de las particiones difusas de las variables de salida— a estos subespacios según un criterio de optimalidad (esto es, siguiendo la metodología COR).

Por tanto, de acuerdo con la figura 1, cada nodo $S_h \in S^+$ se asigna a cada consecuente candidato $(B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)$ y al símbolo especial "no considerar" ($R_\emptyset$) que indica la ausencia de reglas en el correspondiente subespacio.

Información heurística La información heurística sobre la preferencia potencial de elegir un vector de consecuentes específico, B^{k_h}, en cada combinación de antecedentes (subespacio) se determina de la siguiente forma para $B^{k_h} = (B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)$:

$$\eta_{hk_h} = \max_{e_l \in E^+(S_h)} Min\left(\mu_{A^h}(x^l), \mu_{B_j^{k_h}}(y^l)\right).$$

Si $B^{k_h} \notin C(S_h)$, hacer $\eta_{hk_h} = 0$.

Posteriormente, para el símbolo "no considerar", hacer lo siguiente:

$$\eta_{h,|\mathcal{B}_1|\cdot\ldots\cdot|\mathcal{B}_m|+1} = \frac{1}{\max\limits_{k_h \in \{1,\ldots,|C(S_h)|\}} \eta_{hk_h}}.$$

Inicialización de la feromona El valor inicial de la feromona para cada asignación se obtiene de la siguiente forma:

$$\tau_0 = \frac{1}{|S^+|} \sum_{S_h \in S^+} \max_{B^{k_h} \in C(S_h)} \eta_{hk_h}.$$

Así, la feromona inicial será el valor medio del camino construido tomando el mejor consecuente en cada regla de acuerdo con la información heurística (se trata, por tanto, de una asignación voraz).

Esquema de OCH: sistema de la mejor-peor hormiga Una vez diseñados los componentes previos, se debe definir un algoritmo OCH para resolver el problema. En esta contribución consideramos el algoritmo Sistema de la Mejor-Peor Hormiga (SMPH) [3], que ha demostrado tener un buen comportamiento en varios problema de optimización. Su esquema global se muestra en la figura 2, para más detalle refiérase a [3].

Entradas:

- *Un conjunto de datos entrada-salida* —$E = \{e_1, \ldots, e_l, \ldots, e_N\}$, donde $e_l = (x_1^l, \ldots, x_n^l, y_1^l, \ldots, y_m^l)$, $l \in \{1, \ldots, N\}$, siendo N el tamaño del conjunto de datos, y n (m) el número de variables de entrada (salida)— *que represente el comportamiento del problema a resolver.*
- *Una partición difusa de los espacios de las variables.* En nuestro caso, consideramos conjuntos difusos con distribución equidistante. Sea $\mathcal{A}_i$ el conjunto de términos lingüísticos de la i-ésima variable de entrada, siendo $i \in \{1, \ldots, n\}$, y $\mathcal{B}_j$ el conjunto de términos lingüísticos de la j-ésima variable de salida, siendo $j \in \{1, \ldots, m\}$, donde $|\mathcal{A}_i|$ ($|\mathcal{B}_j|$) es el número de etiquetas de la i-ésima (j-ésima) variable de entrada (salida).

Algoritmo:

1. *Construcción del espacio de búsqueda*:

   1.1. *Definir los subespacios de entrada difusos que contengan ejemplos positivos*: Para ello, debemos definir el conjunto de ejemplos positivos ($E^+(S_s)$) para cada subespacio de entrada difuso $S_s = (A_1^s, \ldots, A_i^s, \ldots, A_n^s)$, donde $A_i^s \in \mathcal{A}_i$ es una etiqueta, $s \in \{1, \ldots, N_S\}$, y $N_S = \prod_{i=1}^{n} |\mathcal{A}_i|$ es el número de subespacios de entrada difusos. En este artículo realizamos lo siguiente:

   $$E^+(S_s) = \{e_l \in E \; tal \; que \; \forall i \in \{1, \ldots, n\}, \forall A_i' \in \mathcal{A}_i, \; \mu_{A_i^s}(x_i^l) \geq \mu_{A_i'}(x_i^l)\},$$

   siendo $\mu_{A_i^s}(\cdot)$ la función de pertenencia asociada a la etiqueta A_i^s.
   Entre todos los N_S subespacios de entrada difusos posibles, considerar sólo aquellos que contengan al menos un ejemplo. Para ello, el conjunto de subespacios con ejemplos positivos se define como $S^+ = \{S_h \mid E^+(S_h) \neq \emptyset\}$.

   1.2. *Generar el conjunto de reglas candidatas en cada subespacio con ejemplos positivos*: En primer lugar se define el conjunto de consecuentes candidatos asociado con cada subespacio que contenga al menos un ejemplo, $S_h \in S^+$. En este artículo realizamos lo siguiente:

   $$\begin{aligned} C(S_h) = \{ \quad & (B_1^{k_h}, \ldots, B_m^{k_h}) \in \mathcal{B}_1 \times \ldots \times \mathcal{B}_m \; tal \; que \\ & \exists e_l \in E^+(S_h) \; donde \; \forall j \in \{1, \ldots, m\}, \forall B_j' \in \mathcal{B}_j, \; \mu_{B_j^{k_h}}(y_j^l) \geq \mu_{B_j'}(y_j^l) \quad \}. \end{aligned}$$

   A continuación se define el conjunto de reglas candidatas para cada subespacio como $CR(S_h) = \{R_{k_h} = [\text{SI } X_1 \text{ es } A_1^h \text{ y ... y } X_n \text{ es } A_n^h \text{ ENTONCES } Y_1 \text{ es } B_1^{k_h} \text{ y ... y } Y_m \text{ es } B_m^{k_h}] \; tal \; que \; (B_1^{k_h}, \ldots, B_m^{k_h}) \in C(S_h)\}$.
   Para permitir a la metodología COR reducir el número inicial de reglas difusas, se añade el elemento especial $R_\emptyset$ (que significa "no considerar") a cada conjunto de reglas candidatas, es decir, $CR(S_h) = CR(S_h) \cup R_\emptyset$. Si se selecciona este elemento, no se generará ninguna regla en el correspondiente subespacio de entrada difuso.

2. *Selección del conjunto de reglas difusas más cooperativo* — Esta fase se realiza ejecutando un algoritmo de búsqueda combinatoria para encontrar la combinación $RB = \{R_1 \in CR(S_1), \ldots, R_h \in CR(S_h), \ldots, R_{|S^+|} \in CR(S_{|S^+|})\}$ con el mejor comportamiento. Dado que el espacio de búsqueda manejado suele ser grande, se deben usar técnicas de búsqueda aproximadas.

   Se considera un índice $f(RB)$ para medir la calidad global del conjunto de reglas codificada en cada solución. En nuestro caso, el algoritmo usa la función de error global basada en el *error cuadrático medio* (ECM), definido como

   $$\text{ECM} = \frac{1}{2 \cdot N} \sum_{l=1}^{N} (\mathbf{F}(x_1^l, \ldots, x_n^l) - \mathbf{y}^l)^2,$$

   donde $\mathbf{F}(x_1^l, \ldots, x_n^l)$ es el vector de salida obtenido a partir del SBRD cuando se usa el ejemplo e_l como entrada, y $\mathbf{y}^l$ es el vector de salida deseado. Cuanto más cercano a cero sea la medida, mejor comportamiento global y, por tanto, mejor cooperación entre reglas.

Figura 1: Algoritmo COR

3 MÉTODO DE APRENDIZAJE DE LA BC DE UN SBRDL

En este apartado describiremos nuestra propuesta para el aprendizaje automático de la BC de un SBRDL, compuesta por dos mecanismos:

- Un proceso de aprendizaje de la BD mediante un Algoritmo Genético (AG) que nos va a definir:
  - Dominio de cada variable que interviene en el modelo.
  - Número de etiquetas de cada variable.
  - Semántica de cada etiqueta considerando particiones difusas no uniformes. Para

1. Dar un valor inicial de la feromona, τ_0, para cada arco.
2. Mientras (*condición_terminación* no se cumpla) hacer:
   (a) Realizar el camino de cada hormiga mediante el **proceso de construcción de la solución**.
   (b) Aplicar el **mecanismo de evaporación de feromona**.
   (c) Aplicar el **proceso de búsqueda local** sobre la mejor solución actual.
   (d) Actualizar $S_{mejor\ global}$ y $S_{peor\ actual}$.
   (e) Aplicar la **regla de actualización de feromona Mejor-Peor**.
   (f) Aplicar la **mutación de feromona**.
   (g) Si (*condición_estancado* se cumple), aplicar **reinicialización**.

Figura 2: Algoritmo SMPH

construir dichas particiones se emplea una función de escala no lineal que define áreas con mayor o menor "sensibilidad".

- el algoritmo e aprendizaje de reglas COR-OCH, que deriva la BR tomando como punto de partida la BD previamente obtenida en el paso anterior. De esta manera, el método híbrido propuesto obtiene una definición completa de la BC mediante la acción cooperativa de ambos procesos.

Ya que todos los componentes de la BD irán evolucionando mediante un AG, sería aconsejable reducir la dimensión del espacio de búsqueda. Por tanto, el uso de una función de escala no lineal debería estar condicionado a escoger una función parametrizada con un reducido número de parámetros. Para este trabajo, hemos considerado la función propuesta en [8], que solamente tiene un parámetro denominado a ($a \in \mathbb{R}$). La función es: $f : [-1, 1] \rightarrow [-1, 1]$

$$f(x) = sign(x) \cdot |x|^a, \quad con\ \ a > 0$$

El resultado final es un valor entre $[-1, 1]$. La acción del parámetro a en la partición difusa depende de su valor: sensibilidad uniforme ($a = 1$), mayor sensibilidad en los valores centrales ($a > 1$), o mayor sensibilidad en los valores extremos ($a < 1$). En este trabajo se han considerado funciones de pertenencia triangulares, por tanto, la función de escala no lineal sólo se aplica a los tres puntos que definen la función de pertenencia. Es importante destacar que esa función produce efectos simétricos alrededor del punto central del dominio. Con una pequeña modificación se puede generar mayor sensibilidad únicamente en uno de los extremos (el proceso completo de construcción de la partición difusa se puede consultar en [4]). La figura 3 muestra una representación gráfica de esas cinco posibilidades para la partición difusa.

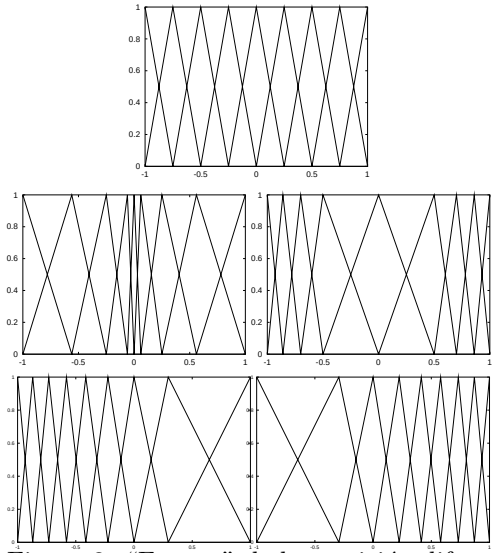

Figura 3: "Formas" de la partición difusa

A continuación, se describirá el proceso evolutivo que monitoriza el aprendizaje de la BC. Los AGs[9] son técnicas de optimización y búsqueda basadas en una formalización del proceso evolutivo de los seres vivos. Cuando se trabaja con AGs, los aspectos importantes son cómo codificar cada solución (en este caso, la BD de un SBRDL), como evaluar esa solución y como crear nuevas soluciones:

Codificación de la BD. Para un modelo con N variables, cada individuo o cromosoma está compuesto de cuatro partes:

- Número de etiquetas (C_1): codificados en un vector de valores enteros de longitud N. Para este trabajo, el rango considerado es $E \in \{2, \ldots, 7\}$.
- Parámetros de "forma": (C_2): codificados en un vector de enteros de longitud N, que puede tomar los valores $S \in \{0, 1, 2, 3, 4\}$, que representan las cinco posibilidades mostradas en la figura 3.
- Parámetros de sensibilidad (C_3): codificados en un vector de valores reales de longitud N, donde se almacena el valor del parámetro (a) de la función de escala no lineal para cada variable. El rango considerado es el intervalo $(0, 10)$.
- Dominios (C_4): codificados en un vector de $N \times 2$ valores reales que almacenan el dominio de cada variable ($[v_{min}, v_{max}]$). Si el dominio inicial de una variable es $[v^i_{inf}, v^i_{sup}]$, y d es la dimensión del intervalo ($d = v^i_{sup} - v^i_{inf}$), el rango para el límite inferior del dominio es $[v^i_{inf} - (1/4 * d), v^i_{inf}]$, y el rango para el límite superior es $[v^i_{sup}, v^i_{sup} + (1/4 * d)]$.

Una representación gráfica del cromosoma sería:

$$C_1 = (E_1, \ldots, E_N) \qquad C_2 = (S_1, \ldots, S_N)$$

$$C_3 = (a_1, \ldots, a_N) \quad C_4 = (v^1_{min}, v^1_{max}, \ldots, v^N_{min}, v^N_{max})$$

Evaluación de los cromosomas. El cálculo de la función de evaluación consta de tres pasos:

- Construir las particiones difusas de cada variable, utilizando la información contenida en el cromosoma. Primero, el dominio de cada variable (C_4) es normalizado al intervalo $[-1, 1]$. A continuación, se crea una partición difusa uniforme considerando su número de etiquetas (C_1). Después, la función de escala no lineal, con su "forma" (C_2) y su parámetro de sensibilidad (C_3) se aplica a los tres puntos que definen cada etiqueta. Finalmente, la partición es trasladada de nuevo a su dominio original (C_4).
- Generar la BR mediante COR-OCH, considerando la BD obtenida en el paso previo.
- Calcular el error cuadrático medio (ECM) sobre el conjunto de datos de entrenamiento usando la BC obtenida (BD + BR). Para mejorar la capacidad de generalización del SBRD resultante, intentando evitar el posible sobreaprendizaje, penalizaremos los SBRDLs con un número de reglas (NR) excesivo. Así, la función de evaluación es:

  $$F = \omega_1 \cdot ECM + \omega_2 \cdot NR$$

  siendo ω_1 y ω_2 porcentajes de peso de cada componente ($\omega_1 + \omega_2 = 1$). Para los experimentos, hemos considerado $\omega_1 = 0.9$.

Operadores genéticos. Como operador de selección, hemos utilizado el muestreo universal estocástico. Respecto a los operadores de recombinación genética, y dada la especial estructura de los cromosomas, con cuatro partes fuertemente relacionadas, es necesario diseñar operadores específicos que actúen de forma coordinada. Se han utilizado dos operadores de mutación, uno para codificación entera (C_1 y C_2) y otro para codificación real (C_3 y C_4). para el primer caso, se emplea el esquema propuesto por Thrift en [10] y para el segundo se utiliza el operador de mutación no uniforme de Michalewicz [9].

Respecto al operador de cruce, se emplean dos procesos diferentes dependiendo de las características de los cromosomas implicados. Si ambos progenitores tienen los mismos valores en C_1 y C_2 (para todas las variables), el AG ha localizado una zona del espacio de búsqueda que debe ser explotada. Para eso, aplicamos el operador de cruce *parent centric* [6] en C_3 y C_4 manteniendo los valores de C_1 y C_2 en los descendientes. En el otro caso, se utiliza un operador de cruce clásico en un punto sobre las cuatro partes del cromosoma. Es decir, se selecciona aleatoriamente un punto de cruce p ($p \in \{1, \ldots, N\}$) y los dos padres se cruzan por la variable p en C_1, C_2, C_3 y C_4, produciendo dos descendientes que mantienen la correlación que había entre los distintos componentes de cada variable en los progenitores.

4 EXPERIMENTOS

El método propuesto se ha probado con un problema de estimación del coste de mantenimiento de la red eléctrica de una ciudad, basado en datos simulados de la red óptima para 1059 ciudades, con cuatro variables de entrada y una de salida. Para los experimentos, se ha realizado una validación cruzada con 5 particiones distintas. Es decir, el conjunto de datos se divide en 5 subconjuntos de igual tamaño (aproximadamente) y el método es ejecutado 5 veces, dejando en cada una de ellas uno de esos subconjuntos como conjunto de datos de prueba y utilizando los otros cuatro como conjunto de datos de entrenamiento[1]. Para cada una de las cinco particiones de datos, se ha ejecutado el AG seis veces con diferentes valores para la semilla de inicio de la secuencia de números aleatorios. Los parámetros del AG son: tamaño de población= 64, probabilidad de cruce= 0.6, probabilidad de mutación= 0.1 y condición de parada= 2000 evaluaciones.

Los resultados obtenidos por el método propuesto (AG+COR-OCH) se han comparado con otros dos mecanismos: uno de ellos comprende el AG propuesto en este trabajo pero utilizando el método de Wang y Mendel [11] para aprender la BR (AG+WM) y el otro considera únicamente el método COR-OCH con una partición difusa uniforme y el mismo número de etiquetas en todas las variables (5 etiquetas, al ser el valor con el que se ha obtenido el modelo más preciso).

Los resultados se muestran en la tabla 1. Las columnas numeradas de 1 a 5 en la primera fila representan las cinco particiones utilizadas. En cada una de esas columnas aparecen los valores medios de las seis ejecuciones del AG realizadas. Las últimas tres columnas representan: la media aritmética ($\bar{x}$) sobre las cinco columnas anteriores, la varianza de esos mismos cinco valores ($\sigma_{\bar{x}_i}$) y la media aritmética de las varianzas obtenidas entre las seis ejecuciones de cada partición ($\bar{\sigma_{x_i}}$). Mientras que $\sigma_{\bar{x}_i}$ representa las diferencias entre las cinco particiones, $\bar{\sigma_{x_i}}$ representa las diferencias entre las distintas ejecuciones de una misma partición. Es decir, el primer valor es una medida de la dependen-

[1] los conjuntos de datos utilizados se pueden obtener en `http://decsai.ugr.es/~casillas/FMLib/`

Tabla 1: Resultados obtenidos

		1	**2**	**3**	**4**	**5**	$\bar{x}$	$\sigma_{\bar{x}_i}$	$\bar{\sigma_{x_i}}$
COR-OCH	ECM_{tra}	38483	39106	39851	40746	40013	**39640**	966	566
	ECM_{tst}	46452	46050	40689	35850	39373	**41683**	4397	1599
	$\#R$	42.3	40.0	40.5	40.2	42.0	**41.0**	1.8	1.4
	$\#E$	25	25	25	25	25	**25**	-	-
AG+COR-OCH	ECM_{tra}	19834	20678	20631	20640	20018	**20360**	402	1561
	ECM_{tst}	25076	24653	20919	23605	19899	**22830**	2302	3259
	$\#R$	18.3	17.8	15.2	17.7	19.8	**17.8**	1.7	3.8
	$\#E$	22.3	22.8	20.2	21.5	22.5	**21.9**	1.1	1.8
GA+WM	ECM_{tra}	22765	22682	24147	21697	23778	**23014**	971	2143
	ECM_{tst}	25826	23942	23817	24319	22549	**24090**	1176	3667
	$\#R$	54.0	44.5	51.0	54.2	52.0	**51.1**	3.9	6.8
	$\#E$	26.3	25.0	26.0	27.2	26.5	**26.2**	0.8	1.5

cia que tiene el método de la partición considerada, y el segundo mide cuán robusta es la metaheurística para obtener resultados similares independientemente de la semilla utilizada. Los cuatro valores que se presentan en cada columna para cada método son los ECM sobre entrenamiento y prueba ($\mathbf{ECM_{tra}}, \mathbf{ECM_{tst}}$), el número de reglas de la BR ($\#\mathbf{R}$) y la suma del número de etiquetas de cada variable ($\#\mathbf{E}$).

Como se puede observar en la tabla, el método propuesto mejora claramente a los otros dos, obteniendo valores de predicción más precisos (tanto en entrenamiento como en prueba) y modelos más interpretables (menor número de reglas y de etiquetas). Por otro lado, AG+COR-OCH y AG+WM son ligeramente menos dependientes a la partición utilizada ($\sigma_{\bar{x}_i}$) que COR-OCH, mientras que éste último es menos dependiente de la semilla de inicio ($\bar{\sigma_{x_i}}$) que los anteriores.

El principal inconveniente del mecanismo propuesto es el tiempo requerido por el AG. Cada vez que se genera un nuevo cromosoma, es necesaria una ejecución de COR-OCH. Aunque dicho proceso no es excesivamente costoso en tiempo, para algunos problemas complejos (con muchas variables), sí puede suponer un inconveniente. En trabajos futuros intentaremos ajustar la estructura y parámetros del AG para reducir su tiempo de ejecución manteniendo en lo posible su buen comportamiento.

Referencias

[1] J. Casillas, O. Cordón, F. Herrera, COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules, *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 32(4), 2002, pp. 526-537.

[2] J. Casillas, O. Cordón, F. Herrera. COR methodology: a simple way to obtain linguistic fuzzy models with good interpretability and accuracy. En J. Casillas, O. Cordón, F. Herrera, L. Magdalena (Eds.), Accuracy improvements in linguistic fuzzy modeling, (Studies in Fuzziness and Soft Computing, Springer, 2003), págs. 27–45.

[3] O. Cordón, F. Herrera, I. Fernández de Viana, L. Moreno, A new ACO model integrating evolutionary computation concepts: the best-worst ant system, *Proceedings of the 2nd International Workshop on Ant Algorithms*, 2000, pp. 22-29.

[4] O. Cordón, F. Herrera, L. Magdalena, P. Villar, A genetic learning process for the scaling factors, granularity and contexts of the fuzzy rule-based system data base, *Information Science* 136, 2001, pp. 85-107.

[5] O. Cordón, F. Herrera, P. Villar, Analysis and guidelines to obtain a good uniform fuzzy partition granularity for fuzzy rule-based systems using simulated annealing, *International Journal of Approximate Reasoning* 25, 2000, pp. 187-215.

[6] K. Deb, D. Joshi, A. Anand, Real-Coded Evolutionary Algorithms with Parent-Centric Recombination, *Proc. of the 2002 congress on evolutionary computation*, IEEE press, 2002, pp. 61-66.

[7] M. Dorigo, G. Di Caro. The ant colony optimization meta-heuristic. En D. Corne, M. Dorigo, F. Glover (Eds.), New ideas in optimization (McGraw-Hill, 1999), págs. 11–32.

[8] L. Magdalena, Adapting the gain of an FLC with genetic algorithms, *International Journal of Approximate Reasoning* 17(4), 1997, pp. 327-349.

[9] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs (Springer-Verlag, 1996).

[10] P. Thrift, Fuzzy logic synthesis with genetic algorithms. Proceedings of Fourth International Conference on Genetic Algorithms (ICGA'91), 1991, pp. 509-513.

[11] L.X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6), 1992, pp. 1414–1427.