Un esquema de pesos basado en evolución diferencial para generación de prototipos

Isaac Triguero[1], Joaquín Derrac[1], Salvador García[2], and Francisco Herrera[1]

[1] Dept. de Ciencias de la Computación e Inteligencia Artificial, CITIC-UGR.
Universidad de Granada. 18071, Granada, España
triguero@decsai.ugr.es, jderrac@decsai.ugr.es, herrera@decsai.ugr.es
[2] Dept. de Informática. Universidad de Jaén. 23071, Jaén, España
sglopez@ujaen.es

Resumen Las técnicas de generación de prototipos han demostrado ser muy competitivas mejorando el rendimiento del clasificador del vecino más cercano. Dentro de la metodología de generación de prototipos, cabe destacar los métodos de posicionamiento de prototipos como aquellos que poseen una mayor eficacia. En la literatura se han usado distintos algoritmos evolutivos para optimizar la posición de los prototipos con resultados prometedores. No obstante, estos resultados pueden ser mejorados si otras técnicas de reducción de datos, tales como los esquemas de pesos, son consideradas.
En este trabajo se propone un enfoque híbrido para integrar un esquema de pesos con una de las técnicas más prometedoras en generación de prototipos. Concretamente, se aplicará una técnica auto-adaptativa de evolución diferencial con el fin de optimizar la ponderación dada a cada característica y la localización de los prototipos. Los resultados obtenidos, contrastados mediante técnicas estadísticas, avalan la utilidad de este enfoque híbrido para mejorar el rendimiento del clasificador del vecino más cercano.

Keywords: Generación de prototipos, evolución diferencial, esquema de pesos, clasificación

1. Introducción

La regla del vecino más cercano (*Nearest Neighbor*, NN) [1] es una técnica de clasificación supervisada, caracterizada por su simplicidad y efectividad, que pertenece a la familia de los métodos de aprendizaje basado en instancias [2]. El NN es un clasificador no paramétrico que requiere que todos los datos de entrenamiento estén almacenados. Los ejemplos no conocidos son clasificados en base a la clase de las instancias más cercanas a ellos, en términos de distancia. A pesar de su simplicidad y eficacia, esta técnica tiene algunas debilidades como el tiempo de respuesta, la tolerancia al ruido y el coste de almacenamiento. Además, este clasificador realiza sus predicciones en base a los datos de entrada, asumiendo que estos definen perfectamente la distribución de clases.

Estas debilidades han sido abordadas por diferentes enfoques. Entre ellos, una buena solución que podemos encontrar en la literatura consiste en reducir el conjunto de datos

usados para clasificar. Estas técnicas tratan de reducir el conjunto de entrenamiento, con el objetivo de eliminar ejemplos ruidosos, redundantes e irrelevantes. Desde el punto de vista del espacio de las características/atributos, podemos destacar la selección de características [4] como una de las técnicas más importantes, que consiste en elegir un subconjunto representativo de atributos. Por otro lado, desde el punto de vista de las instancias, los métodos de reducción de datos se dividen en dos enfoques, conocidos como selección de prototipos (SP) [5] y generación/abstracción de prototipos (GP) [8] . El primero consiste en seleccionar un subconjunto de los datos de entrenamiento originales, mientras que la GP puede generar nuevos datos artificiales si los necesita. De este modo, la GP no supone que los datos originales delimitan perfectamente las fronteras de decisión entre clases.

Otra propuesta interesante es el empleo de esquemas de pesos aplicados a las características (*Feature Weighting*, FW) [10] que permite asignar un peso a cada característica del dominio del problema para modificar el modo en que las distancias entre ejemplos son calculadas. Esta técnica puede verse como una generalización de los métodos de selección de características, permitiendo dar un grado de relevancia a cada atributo del problema mediante un valor real.

En los últimos años han aparecido una gran cantidad de propuestas evolutivas aplicadas a problemas de minería de datos. Debido a que los problemas de GP y FW pueden ser vistos como problemas de optimización continua, también pueden ser abordados mediante algoritmos evolutivos. Los algoritmos evolutivos para GP se basan en técnicas de ajuste del posicionamiento de prototipos [6], que es una metodología adecuada para optimizar la localización de los prototipos. Específicamente, el algoritmo de Evolución Diferencial (ED) [11] y sus propuestas avanzadas [12] han demostrado ser las técnicas de posicionamiento más efectivas. Respecto a los métodos de FW,se ha propuesto muchas técnicas evolutivas aplicadas al algoritmo NN.

Habitualmente, los métodos de ajuste de la localización de los prototipos [6] se centran en el proceso de optimizar la posición, sin tomar en consideración la selección más apropiada del número de prototipos por clase. Recientemente, en [7,9], este problema se aborda con un proceso de adición iterativo, en el que se determina qué clases necesitan más prototipos para ser representada. Este algoritmo se conoce con el nombre de IPADECS.

En este trabajo, se propone un método híbrido que combina IPADECS con un esquema de FW. En este esquema, los pesos de cada característica y la localización de los prototipos se determinan mediante un método auto-adaptativo de ED, conocido con el nombre de SFLSDE [12]. Este método se emplea para abordar los problemas de FW y GP simultaneamente. Los algoritmos de GP evolutivos tienden a sobre-entrenar los datos de entrenamiento, en pocas iteraciones, por lo que la modificación de la función de distancia mediante FW puede evitar este problema, determinando además un grado de relevancia para cada característica. Tras su descripción, se presenta un estudio experimental en el que se muestran las mejoras significativas en comparación con la técnica original de GP y otras técnicas de FW.

El resto del trabajo se organiza como sigue: La Sección 2 presenta algunos conceptos preliminares sobre las técnicas empleadas. La Sección 3 describe el modelo propuesto.

La Sección 4 define el estudio experimental y presenta los resultados. Por último, la Sección 5 concluye el trabajo.

2. Preliminares

Esta sección repasa algunos conceptos preliminares necesarios. La Sección 2.1 define formalmente el problema de la generación de prototipos. La Sección 2.2 comenta el empleo de esquemas de pesos aplicados a las características. Finalmente, la Sección 2.3 explica el algoritmo de ED.

2.1. Generación de prototipos

La generación de prototipos [8] tiene como objetivo obtener un conjunto de entrenamiento lo más reducido posible que permita al NN clasificar con la misma o más calidad que con el conjunto de entrenamiento original. Esto permite reducir la complejidad espacial del método y reducir su coste computacional. Además, en ocasiones puede mejorar su precisión, mediante la eliminación de ruido.

Su definición formal es la siguiente: Sea $\mathbf{X}$ una instancia donde $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M, \mathbf{c})$, con $\mathbf{X}$ perteneciendo a la clase $\mathbf{c}$ y un espacio M-dimensional donde $\mathbf{x}_i$ es el i-ésimo valor de la instancia X. Asumamos que existe un conjunto de entrenamiento CE compuesto por N instancias, y un conjunto de test CT compuesto por T instancias. El propósito de la GP es obtener un conjunto generado de prototipos (CGP), que consiste en r, $r < N$, prototipos $\mathbf{P}$ donde $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_M, \mathbf{c})$, que son generados a partir de los ejemplos de CE. El CGP debe representar eficientemente la distribución de clases y su tamaño debe ser lo suficientemente pequeño para mejorar el tiempo y necesidad de almacenamiento del clasificador NN.

2.2. Esquemas de pesos

El objetivo de los métodos de FW es reducir la sensibilidad a características redundantes, irrelevantes o ruidosas en el clasificador NN, mediante la modificación de la medida de distancia con pesos, para mejorar su precisión en clasificación.

Normalmente se usa la distancia euclídea (Ecuación 1) en el clasificador NN, donde P and Q son dos ejemplos y M es el número de características de estos. Durante este estudio se usará dicha medida, dada su simplicidad, facilidad para optimizarla y su amplio uso en el campo del aprendizaje basado en instancias [2].

$$DistanciaEuclidea(P,Q) = \sqrt{\sum_{i=1}^{M}(p_i - q_i)^2} \tag{1}$$

Lo métodos de FW suelen extender dicha ecuación aplicando diferentes pesos a cada característica (W_i) que modifica la forma en que la medida de distancia es calculada (Ecuación 2).

$$DistanciaFW(P,Q) = \sqrt{\sum_{i=1}^{M} W_i \cdot (p_i - q_i)^2} \tag{2}$$

Esta técnica ha sido ampliamente usada en la literatura. En [10] podemos encontrar un estudio completo sobre dichos métodos.

2.3. Evolución diferencial

El algoritmo de ED comienza con una población de NP individuos. La población inicial debe cubrir el espacio de búsqueda tanto como sea posible. En algunos problemas esto se consigue inicializando los individuos aleatoriamente, pero en otros problemas, como en la GP, existe un conocimiento base que está disponible para ser usado en los mecanismos de inicialización. Las generaciones siguientes se denotan como: $G = 0, 1, \ldots, G_{max}$. En ED es usual denominar a cada individuo como un vector D-dimensional $X_{i,G} = \{x^1_{i,G}, ..., x^D_{i,G}\}$, llamado vector objetivo.

Después de la inicialización, el algoritmo ED aplica un operador de mutación para generar un vector mutado $V_{i,G}$ respecto a cada individuo $X_{i,G}$, en la población actual. Para cada vector objetivo $X_{i,G}$, su vector mutado asociado es $V_{i,G} = \{V^1_{i,G}, ..., V^D_{i,G}\}$.

En este trabajo nos centramos en el uso de la estrategia *RandToBest/1* que genera un vector mutado de la siguiente forma:

$$V_{i,G} = X_{i,G} + F \cdot (X_{mejor,G} - X_{i,G}) + F \cdot (X_{r^i_1,G} - X_{r^i_2,G}) \tag{3}$$

Donde los índices r^i_1, r^i_2 son enteros mutuamente exclusivos y generados aleatoriamente en el rango $[1, NP]$, y son también diferentes del índice base i. El factor de escala F es un parámetro de control positivo para escalar la diferencia entre vectores.

Tras la fase de mutación, el operador de cruce se aplica para cada par, vector objetivo $X_{i,G}$ y su correspondiente vector mutado $V_{i,G}$, para generar un nuevo vector, denominado vector de "prueba" $U_{i,G}$. Nos centraremos en el uso del operador de cruce binomial, inicializado con los valores del vector objetivo, y en el cual cada componente del vector de prueba se modifica con los valores del vector mutado si al generar un número aleatorio entre 0 y 1, éste es menor o igual que un ratio de cruce RC. Finalmente, debemos decidir qué individuo pasa a la siguiente generación $G + 1$. Si el vector de prueba obtiene una solución igual o mejor que el vector objetivo, este reemplaza su correspondiente vector objetivo en la siguiente generación; en otro caso, el vector objetivo se mantiene en la población.

3. Un modelo híbrido que integra un esquema de pesos con generación de prototipos

En esta sección se describe el modelo híbrido propuesto y sus principales componentes. En la Sección 3.1 se presenta el modelo propuesto como esquema de pesos. A continuación, la Sección 3.2 presenta el modelo híbrido en detalle.

3.1. Evolución diferencial como un esquema de pesos

Como se dijo anteriormente, los esquemas de pesos aplicados a las características se pueden entender como un problema de búsqueda en un espacio continuo en el cual

necesitamos determinar la ponderación más apropiada para cada característica con el fin de mejorar el clasificador NN. Específicamente, se propone el uso de un algoritmo de ED para obtener los mejores pesos que permitan, a un conjunto reducido CGP, incrementar su precisión en clasificación sobre el conjunto CE. Denominamos a este algoritmo con las siglas "EDFW" (Evolución Diferencial para FW).

El EDFW comienza con una población de NP individuos $X_{i,G}$. Este algoritmo codifica un vector de pesos en cada individuo, como un vector M-dimensional de números reales, que contiene la ponderación de cada característica mediante un valor comprendido en en el intervalo [0,1]. Esto implica, que cada individuo de la población codifica una solución completa para el problema de FW. Siguiendo las ideas establecidas en [12], la población inicial se genera de forma aleatoria dentro del rango definido anteriormente.

Después del proceso de inicialización, el metodo EDFW entra en un bucle en el cual los operadores de mutación y cruce guían el proceso de optimización de los pesos en las características generando nuevos vectores $U_{i,G}$. Tras aplicar dichos operadores, como se explicaba en la Sección 2.3, debemos verificar si los valores generados pertenecen al rango $[\theta, 1]$. Si el valor es mayor que uno, este se truncará a uno. Además, basándonos en el trabajo de Kira y Rendell [15], si este valor es menor que un umbral θ, se considera que esta característica es irrelevante, y por tanto se establece a 0. En nuestros experimentos, este umbral se ha establecido empíricamente en 0,2.

Por último, el operador de selección decide si el vector generado debe entrar en la siguiente generación $G+1$. La regla del NN guía este operador para cubrir nuestros propósitos. Los ejemplos de CE son clasificados con los prototipos de un CGP dado, pero en este caso, la medida de distancia usada en el clasificador NN es modificada de acuerdo con la Ecuación 2, donde los pesos W_i son obtenidos de $X_{i,G}$ y $U_{i,G}$. La precisión o acierto en clasificación se mide como el número de éxitos de clasificación dividido entre el número total de clasificaciones realizadas. Nuestro objetivo es maximizar dicho valor, por ello, el operador de selección se puede ver como:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } Precision(CGP, U_{i,G}) \\ & >= Precision(CGP, X_{i,G}) \\ X_{i,G} & \text{En otro caso} \end{cases} \quad (4)$$

En caso de empate entre los valores de precisión obtenidos, seleccionamos el vector $U_{i,G}$ con el fin de dar al individuo mutado la posibilidad de entrar en la población. Para evitar que los parámetros mas importantes en ED, (F y RC), provoquen una convergencia lenta o prematura, usamos las ideas propuestas en [12] para implementar un esquema auto-adaptativo de ED.

3.2. Híbrido IPADECS y esquema de pesos

El algoritmo IPADECS [9] sigue un enfoque iterativo de ajuste del posicionamiento de los prototipos, con un modelo incremental. En cada paso, se aplica un procedimiento de optimización para ajustar la localización de los prototipos, añadiendo nuevos si los necesita. El objetivo de este algoritmo es determinar el número más adecuado de prototipos por clase y ajustar su posición durante el ciclo evolutivo. IPADECS usa el

algoritmo SFLSDE como técnica para optimizar, pero en este caso, cada individuo se codifica como una solución completa, es decir un *CGP*. Al final del algoritmo, éste devuelve el mejor *CGP* encontrado.

El modelo híbrido que compone IPADECS y EDFW puede ser descrito básicamente como la combinación de una etapa IPADECS con una etapa EDFW para determinar los mejores pesos. La Figura 1 muestra el pseudo-código de esta propuesta. Inicialmente, se aplica el algoritmo IPADECS en el que todas las características tienen el mismo grado de relevancia 1.0 (Instrucciones 1-4). La función "EvaluarConPesos" obtiene la precisión en clasificación obtenido al clasificar el CE con el *CGP* como conjunto de entrenamiento y usando los "Pesos" asociados para modificar la función de distancia. Después, este modelo entra en un ciclo en el cual se intenta encontrar la ponderación más apropiada para cada atributo y el mejor posicionamiento para cada prototipo. La instrucción 6 aplica un optimización basada en ED de los pesos de las características, esto es, tomando el mejor *CGP* generado hasta el momento con IPADECS, aplicamos dicho algoritmo para determinar los pesos más apropiados. Los pesos actuales son introducidos como un individuo de la población del método EDFW para asegurar que el método de FW no degrada en ningún momento el rendimiento de *CGP*. A continuación, la instrucción 7 genera un nuevo *CGP*, con IPADECS, pero en este caso, el proceso de optimización toma en consideración los nuevos pesos para calcular las distancias entre prototipos (Ver Ecuación 2). La idea subyacente de esta instrucción es que el algoritmo IPADECS debería generar un *CGP* diferente debido al hecho de que la medida de distancia ha cambiado, por lo tanto, el espacio de búsqueda se ha modificado. Después de este proceso, debemos asegurar que la nueva localización de los prototipos (CGP_{aux}) con sus respectivos pesos han conseguido mejorar la precisión de clasificación respecto de *CGP*. Si el valor calculado para el nuevo CGP_{aux} es mayor que el mejor encontrado hasta el momento, se almacena dicho conjunto como el *CGP* encontrado. De cualquier modo, los pesos obtenidos se almacenan y serán usados en la siguiente iteración para impedir que los resultados puedan empeorar. Finalmente, tras un número previamente fijado de iteraciones, el modelo devuelve el mejor *CGP* con sus respectivos pesos, que está listo para ser usado como conjunto de referencia por el clasificador NN.

4. Estudio experimental y análisis de resultados

Para tratar de caracterizar el rendimiento del híbrido IPADECS-EDFW, se ha realizado un estudio experimental sobre diferentes problemas de clasificación. La Sección 4.1 describe los conjuntos de datos utilizados. La Sección 4.2 enumera los algoritmos de comparación considerados y describe sus parámetros. La Sección 4.3 presenta y analiza los resultados obtenidos, junto con un estudio estadístico para contrastar los resultados.

4.1. Conjuntos de datos

En este estudio, se han empleado 30 conjuntos de datos representando problemas de clasificación. Han sido tomados del repositorio *KEEL-dataset Repository*[3]. La Tabla 1 describe sus principales características: Nombre, número de instancias (#In), número de atributos (#At) y número de clases (#Cl).

[3] `http://www.keel.es/datasets.php`

Figura 1. Híbrido IPADECS-EDFW

```
1: Pesos[1..N] = 1.0
2: mejoresPesos [1..D]= Pesos[1..N]
3: CGP = IPADECS(Pesos);
4: Precision = EvaluarConPesos(CGP, CE, Pesos)
5: for i = 1 to MAXITER do
6:     nuevosPesos[1..D] = EDFW(CGP, Pesos)
7:     CGP_aux = IPADECS(nuevosPesos)
8:     Accuracy_aux = EvaluarConPesos(CGP_aux, CE, nuevosPesos)
9:     if Precision_aux > Precision then
10:        Precision = Precision_aux
11:        CGP = CGP_aux
12:    end if
13:    Pesos = nuevosPesos
14: end for
15: return CGP, Pesos
```

Cuadro 1. Conjuntos de datos utilizados

Conjunto	#In	#At.	#Cl.	Conjunto	#In	#At.	#Cl.
abalone	4.174	8	28	monks	432	6	2
banana	5.300	2	2	newthyroid	215	5	3
bands	539	19	2	nursery	12.690	8	5
breast	286	9	2	pima	768	8	2
bupa	345	6	2	ring	7.400	20	2
chess	3.196	36	2	saheart	462	9	2
cleveland	297	13	5	spambase	4.597	57	2
contraceptive	1.473	9	3	splice	3.190	60	3
dermatology	366	33	6	tae	151	5	3
glass	214	9	7	thyroid	7.200	21	3
housevotes	435	16	2	titanic	2.201	3	2
iris	150	4	3	twonorm	7.400	20	2
lymphography	148	18	4	wisconsin	683	9	2
magic	19.020	10	2	yeast	1.484	8	10
mammographic	961	5	2	zoo	101	16	7

Todos los conjuntos han sido particionados, empleando la técnica de validación cruzada de 10 campos. Además, sus valores han sido normalizados en el intervalo $[0, 1]$, para igualar la influencia de todos los atributos con respecto a la medida de distancia del clasificador.

4.2. Algoritmos y parámetros

Se han empleado tres métodos de comparación para realizar un estudio completo de nuestra propuesta. El primero de ellos es el algoritmo base IPADECS, y dos algoritmos bien conocidos en el ámbito de los esquemas de pesos: TS/KNN [16] y ReliefF [17]. Además, se ha considerado también el clasificador 1-NN como referencia básica. La

Cuadro 2. Parámetros empleados

Método	Parámetros
IPADECS-DEFW	$MAXITER$ = 20, PoblaciónIPADECS = 10, Iteraciones ED = 50 PoblacionEDFW=25 , Iteraciones EDFW=200, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9
IPADECS	Población=10, Iteraciones ED = 500, iterSFGSS=8, iterSFHC=20, Fl=0.1, Fu=0.9
TSKNN	Evaluaciones = 10000, M= 10, N= 2, P= ceil($\sqrt{\#Features}$)
ReliefF	K valor para contribuciones= El mejor en [1,20] Clasificador base: 1-NN

Tabla 2 muestra los parámetros empleados. Aquellos métodos que son estocásticos se han ejecutado tres veces por partición.

4.3. Resultados obtenidos y estudio estadístico

En este estudio experimental, analizamos los resultados en términos de precisión o acierto obtenido durante la clasificación de los datos de test. La Tabla 3 muestra los resultados obtenidos. Para cada conjunto se muestra el valor medio obtenido en cada conjunto por cada algoritmo (Ac.) y su correspondiente desviación típica (Des.). Además, se ha remarcado en negrita el mejor resultado en acierto obtenido en cada conjunto.

Se van a emplear tests no paramétricos de comparaciones múltiples para contrastar los resultados experimentales. Su empleo en minería de datos está recomendado en los casos en que se intenten comparar los resultados de un nuevo algoritmo con respecto a varios métodos simultáneamente [13].

Específicamente, se ha seleccionado el test de Friedman como método para detectar la existencia de diferencias significativas entre los resultados de acierto, y el método de Holm como test *post-hoc* para caracterizar las diferencias encontradas [14] [4].

El test de Friedman detecta diferencias significativas ($p - valor < 10^{-6}$) sobre los resultados. A partir de los rangos obtenidos, se selecciona a IPADECS-EDFW como método de control (aquel que obtiene el rango más bajo), y se aplica el test *post-hoc*. La Tabla 4 resume los resultados del estudio estadístico.

A partir de los resultados, se pueden extraer las siguientes conclusiones:

- IPADECS-EDFW obtiene mejor acierto medio que el resto. Además, obtiene el mejor resultado en 17 de los 30 problemas considerados.
- Comparando IPADECS-EDFW con IPADECS, se aprecia una buena sinergia entre GP y FW, ya que los resultados medios han aumentado considerablemente, siendo un algoritmo más robusto ante diferentes tipos de problemas.
- IPADECS-EDFW mejora estadísticamente los resultados obtenidos por el resto, con un nivel de significancia $\alpha = 0{,}05$. Por tanto, contrastan la afirmación de que la mejora obtenida en precisión de clasificación por IPADECS-EDFW sobre el resto de métodos es significativa.

[4] Más información en el sitio web temático de SCI2S sobre *Statistical Inference in Computational Intelligence and Data Mining* `http://sci2s.ugr.es/sicidm/`

Cuadro 3. Resultados obtenidos

	1-NN		IPADECS		IPADECS-EDFW		TS/KNN		ReliefF	
Conjunto	**Ac.**	**Des.**	**Ac.**	**Des.**	**Ac.**	**Des.**	**Ac.**	**Des.**	**Ac.**	**Des.**
abalone	19,91	1,60	22,21	2,34	**25,47**	2,39	24,65	1,43	14,71	1,85
banana	87,51	1,03	84,09	4,38	**89,70**	0,97	89,51	0,84	68,53	2,76
bands	63,09	4,65	67,15	5,91	69,97	5,89	**73,67**	8,33	70,15	6,38
breast	65,35	6,07	70,91	7,15	71,00	8,52	**72,02**	6,45	62,47	9,71
bupa	61,08	6,88	65,67	8,48	**67,25**	5,27	62,44	7,90	56,46	4,37
chess	84,70	2,36	80,22	3,81	94,52	1,03	95,94	0,40	**96,09**	0,57
cleveland	53,14	7,45	52,49	4,48	54,14	6,20	**56,43**	6,84	55,10	8,62
contraceptive	42,77	3,69	48,54	4,67	**54,79**	3,61	42,70	0,22	39,99	6,05
dermatology	95,35	3,45	96,18	3,01	**96,73**	2,64	96,47	4,01	95,92	2,77
glass	73,61	11,91	69,09	11,13	71,45	11,94	76,42	13,21	**80,65**	12,04
housevotes	92,16	5,41	92,64	3,71	94,00	4,76	**95,16**	3,34	94,00	3,48
iris	93,33	5,16	**94,67**	4,00	**94,67**	4,00	94,00	4,67	94,00	5,54
lymphography	73,87	8,77	78,41	9,31	**80,66**	14,74	74,54	8,95	70,43	22,52
magic	80,59	0,90	80,23	1,47	83,17	1,01	**83,25**	0,68	76,68	5,46
mammographic	73,68	5,59	79,71	4,41	**83,67**	5,55	82,62	4,76	70,76	4,28
monks	77,91	5,42	91,20	4,76	96,10	2,48	**100,00**	0,00	**100,00**	0,00
newthyroid	97,23	2,26	98,18	3,02	**97,71**	3,07	93,48	2,95	97,25	4,33
nursery	82,67	0,92	64,79	4,58	**85,10**	1,42	82,67	0,88	78,94	32,09
pima	70,33	3,53	**76,84**	4,67	71,63	7,35	75,53	5,85	70,32	5,65
ring	75,24	0,82	89,70	1,03	**91,22**	0,94	84,23	1,17	73,08	1,11
saheart	64,49	3,99	70,36	3,07	**71,21**	3,37	68,22	11,35	60,83	9,15
spambase	89,45	1,17	90,89	0,95	92,50	1,38	**92,54**	1,21	60,58	0,08
splice	74,95	1,15	79,78	3,99	**88,53**	1,99	71,72	1,72	78,24	1,30
tae	40,50	8,43	57,71	11,11	**58,33**	12,04	30,54	2,56	49,12	3,77
thyroid	92,58	0,81	93,99	0,36	94,28	0,58	**95,87**	0,61	92,57	0,26
titanic	60,75	6,61	78,19	2,92	**79,01**	2,11	77,78	2,79	61,33	7,90
twonorm	94,68	0,73	97,66	0,69	**97,76**	0,72	96,96	0,87	94,65	1,01
wisconsin	95,57	2,59	**96,42**	1,94	96,28	1,72	96,00	3,61	96,28	2,14
yeast	50,47	3,91	57,35	3,13	**59,17**	3,61	55,86	12,99	51,55	4,97
zoo	92,81	6,57	96,33	8,23	96,67	6,83	66,25	8,07	**96,83**	2,78
Promedio	73,99	19,13	77,39	17,78	**80,22**	17,40	76,92	19,69	73,58	20,39

Cuadro 4. Test estadístico no paramétrico

Método	Rangos	p-valor de Holm
IPADECS-EDFW	1,7500	–
TSKNN	2,5833	0,0412
IPADECS	2,8500	0,0141
ReliefF	3,7333	0,0000
1-NN	4,0833	0,0000

5. Conclusiones

En este trabajo se ha presentado una propuesta de hibridación de generación de prototipos y esquemas de pesos para mejorar la regla del vecino más cercano. De esta forma el modelo híbrido propuesto supera a los métodos de forma aislada. Los resultados obtenidos muestran que esta combinación mejora significativamente al modelo original y a otras propuestas anteriores.

Agradecimientos. Subvencionado por los proyectos TIN2011-28488 y TIC-6858. I. Triguero dispone de una beca FPU de la Universidad de Granada. J. Derrac disfruta de una beca FPU del Ministerio de Educación y Ciencia.

Referencias

1. Cover T.M., Hart P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13, 21–27 (1967).
2. Aha D.W, Kibler D., Albert, M.K.:Instance-Based Learning Algorithms. Machine Learning 1 (6), 37–66 (1991). Springer (2004).
3. Wilson D. R., Martinez, T. R.: Improved Heterogeneous Distance Functions. Journal of Artificial Intelligence Research 6, 1–34 (1997).
4. Liu H., Motoda H. (Eds.):Computational Methods of Feature Selection. Chapman & Hall/Crc Data Mining and Knowledge Discovery Series (2007).
5. García S., Cano J.R., Herrera F.: A memetic algorithm for evolutionary prototype selection: A scaling up approach. Pattern Recognition 41 (8), 2693–2709 (2008).
6. Fernández F., Isasi, P.: Evolutionary Design of Nearest Prototype Classifiers. Journal of Heuristics 10 (4), 431–454 (2004)
7. Triguero I, García S., Herrera F.:IPADE: Iterative Prototype Adjustment for Nearest Neighbor Classification. IEEE Transactions on Neural Networks 21 (12), 1984–1990 (2010).
8. Triguero I, Derrac J, García S., Herrera F.:A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification. IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews, doi: 10.1109/TSMCC.2010.2103939, in press (2011).
9. Triguero I, García S., Herrera F.:Enhancing IPADE Algorithm with a Different Individual Codification. Proceedings of the 6th International Conference on Hybrid Artificial Intelligence Systems (HAIS). LNAI 6679, 262–270 (2011).
10. D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weigthing methods for a class of lazy learning algorithms," Artificial Intelligence Review, vol. 11, pp. 273–314, 1997.
11. Storn R., Price K. V.: A simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11 (10), 341–359 (1997).
12. Ferrante N., Tirronen V.: Scale factor local search in differential evolution. Memetic Computing 1 (2), 153–171 (2009)
13. García S., Herrera F.: An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets"for all Pairwise Comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008).
14. García S., Fernández A., Luengo J., Herrera F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental Analysis of Power. Information Sciences 180, 2044–2064 (2010).
15. Kira K, Rendell L.A.:A practical approach to feature selection. Proceedings of the Ninth International Conference on Machine Learning, Aberdeen, Scotland, 249–256 (1992).
16. Kononenko I.:Estimating attributes: Analysis and extensions of RELIEF. Proceedings of the 1994 European Conference on Machine Learning, Catania, Italy, 171–182 (1994).
17. Tahir M. A., Bouridane A., Kurugollu F.:Simultaneous feature selection and feature weighting using Hybrid Tabu Search/K-nearest neighbor classifier. Pattern Recognition Letters 28 (4) 438-446 (2007)