Búsqueda Dispersa en Selección de Instancias*

Miguel García,** Belén Melián, José A. Moreno,
J. Marcos Moreno Vega, Raquel Rivero

Depto. de Estadística, I.O. y Computación

Escuela Técnica Superior de Ingeniería Informática

Universidad de La Laguna

38271 La Laguna

mgarciat@ull.es, mbmelian@ull.es, jamoreno@ull.es, jmmoreno@ull.es, rrivero@arrakis.es

Resumen

En este trabajo proponemos un algoritmo metaheurístico basado en Búsqueda Dispersa para la resolución del problema de selección de instancias. En una Búsqueda Dispersa un conjunto de soluciones evoluciona mediante mecanismos guiados de diversificación e intensificación. Comparamos nuestra propuesta con otros algoritmos evolutivos propuestos anteriormente. De los resultados obtenidos se concluye que la Búsqueda Dispersa presenta un comportamiento prometedor. Los porcentajes de reducción de instancias, en general, son superiores a los mostrados por los anteriores algoritmos evolutivos, y los porcentajes de acierto similares.

1. Introducción

En un problema de Clasificación, dada una instancia, se desea determinar la clase a la que pertenece dicha instancia. En general, las instancias se representan por un vector de tamaño $n+1$, siendo n el número de variables usadas para caracterizar la instancia. Las n primeras componentes del vector almacenan los valores alcanzados por la instancia en las variables consideradas, y la última almacena la etiqueta de clase de la instancia. La etiqueta de clase es desconocida a priori, siendo el propósito de la clasificación darle valor. La Medicina, la Economía, la Gestión, el Marketing o la Biología son importantes campos en los que aparecen problemas de Clasificación.

Si se dispone de un conjunto de instancias con clase conocida (conjunto de ejemplos de entrenamiento) que puede usarse para determinar el clasificador, se dice que el Aprendizaje es Supervisado. Uno de los clasificadores supervisados más simples es el del *vecino más próximo* [5]. Se trata de un algoritmo de aprendizaje inductivo que clasifica cada nueva instancia por la clase de la instancia más cercana del conjunto de entrenamiento. Para ello es necesario considerar una métrica entre instancias y almacenar el conjunto de ejemplos de entrenamiento. En [17] se proponen varias métricas que pueden usarse con el clasificador del *vecino más próximo*.

Las principales ventajas de este algoritmo de aprendizaje son su sencillez y facilidad de comprensión. El mayor inconveniente es que se deben almacenar todos los ejemplos de entrenamiento, con lo que se disminuye la eficiencia del proceso de clasificación. Además, puede disminuir también la eficacia. Un modo de afrontar estos inconvenientes es llevar a cabo una reducción del conjunto de entrenamiento mediante la selección de instancias.

*Este trabajo ha sido parcialmente financiado por los proyectos TIC2002-04242-C03-01 (70 % son fondos FEDER) y PI042004/088. La actividad desarrollada se enmarca dentro de los objetivos de la red RedHeur (proyecto TIN2004-20061-E).

**Este autor ha sido parcialmente financiado con la beca Cajacanarias

Actas del IV Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB2005, pp.785-792
ISBN: Vol I:84-9732-440-4 Vol II: 84-9732-467-6

En este artículo proponemos una metaheurística Búsqueda Dispersa (BD) [8] para el problema de la reducción de instancias. En una Búsqueda Dispersa, una población inicial de soluciones evoluciona atendiendo a mecanismos guiados de intensificación y diversificación. Desde una población inicial de soluciones, se obtiene un conjunto de referencia formado por *buenas* soluciones. Una solución es *buena* si es de calidad o aporta información no presente en el conjunto de referencia (es dispersa con respecto a las del conjunto de referencia). Del conjunto de referencia se toman subconjuntos de soluciones que se combinan apropiadamente para producir nuevas soluciones. Estas soluciones se usan para actualizar el conjunto de referencia. Estos pasos se reiteran hasta que se cumpla algún criterio de parada (generalmente, mientras cambie el conjunto de referencia).

El artículo está organizado del siguiente modo; en la sección siguiente se introduce el problema de la selección de instancias. En la sección 3 se desarrolla la metaheurística Búsqueda Dispersa aplicada al problema. A continuación se explica la experiencia computacional, y finalmente se exponen las conclusiones.

2. Selección de Instancias

En el problema de selección de instancias se desea encontrar un subconjunto de instancias del conjunto de entrenamiento que, usado para clasificar con el clasificador del vecino más cercano, presente el mejor comportamiento. Para este problema han sido propuestos en la literatura una amplia variedad de métodos de solución. Los métodos propuestos pueden clasificarse en

- métodos basados en reglas del vecino más próximo: Cnn (Condensed Nearest Neighbor) [9], Enn (Edited Nearest Neighbor) [15], Renn (Repited Edited Nearest Neighbor) [15], Vsm (Variable Similarity Metric) [12], Multiedit [6], Mcs (Model Class Selection) [3], Shrink [10], Ib2 (Instance-based Learning Algorithm 2) [10], Ib3 (Instance-based Learning Algorithm 3) [1], Icf (Iterative Case Filtering Algorithm) [2]);
- métodos basados en reglas de eliminación ordenada de instancias: DROP1 (Decremental Reduction Optimization Procedure 1) [16], DROP2 (Decremental Reduction Optimization Procedure 2) [16], DROP3 (Decremental Reduction Optimization Procedure 3) [16];
- métodos basados en muestreo aleatorio: Rmhc(s) (Random mutation hill climbing) [14], Ennrs (Edited Nearest Neighbor by Random Sampling) [18].

En [4] se describen brevemente los anteriores métodos. Asimismo, se proponen cuatro algoritmos de selección de instancias evolutivos (Generational Genetic Algorithm (CGA), Steady-Sate Genetic Algorithm (SGA), CHS Adaptive Search Algorithm y Population-Based Incremental Learning (PBIL)) que son comparados con los anteriores sobre una amplia batería de bases de datos. De los resultados obtenidos en [4] se concluye que los algoritmos evolutivos propuestos presentan un mejor comportamiento que los algoritmos clásicos: los porcentajes de reducción de instancias y de acierto son mayores. Además, CHC fué superior a los otros tres algoritmos evolutivos.

3. Búsqueda Dispersa

Los principios de la Búsqueda Dispersa fueron introducidos en los años 70 como una extensión de ciertas formulaciones para la combinación de reglas de decisión y restricciones de problemas. Esta propuesta inicial permitía generar soluciones teniendo en cuenta las características de diversas partes del espacio de soluciones [7].

La metaheurística de Búsqueda Dispersa [11], [13] es un procedimiento evolutivo basado en poblaciones que genera nuevas soluciones a partir de la combinación de otras, almacenadas en un conjunto de referencia de soluciones, $RefSet$. Las soluciones de referencia se seleccionan a partir de una población de soluciones

distribuidas por todo el espacio de soluciones. El conjunto de referencia es entonces generado atendiendo a determinados criterios de calidad. En las implementaciones que aparecen en la literatura los criterios de calidad citados son tanto calidad según el valor de la función objetivo, que permite incidir en la intensificación del proceso de búsqueda, como calidad en dispersión, que incide en la diversificación del proceso. La evolución del proceso de búsqueda en el espacio de soluciones permite actualizar el conjunto de referencia según convenga. Su actualización se realiza atendiendo a los dos criterios señalados de forma simultánea o únicamente al criterio de calidad del valor de la función objetivo. Una de las características de la búsqueda dispersa es que proporciona un conjunto de buenas soluciones en vez de una única solución heurística tal como sucede con otros procedimientos metaheurísticos.

El procedimiento de Búsqueda Dispersa incluye cinco métodos principales: *Método de Generación de la Diversidad*, que genera soluciones dispersas en todo en el espacio de soluciones; *Método de mejora*, que generalmente permite mejorar una solución dada para obtener un óptimo local; *Método de Actualización del Conjunto de Referencia*, que construye y actualiza el conjunto de referencia de soluciones usadas en el proceso de búsqueda; *Método de Generación de Subconjuntos*, que seleccione los subconjuntos de soluciones del conjunto de referencia que serán combinadas; y *Método de Combinación*, que combina las soluciones de los subconjuntos dados. Una descripción más detallada de los métodos involucrados en la Búsqueda Dispersa se encuentra en [11]. La Figura 1 resume la implementación básica de una Búsqueda Dispersa.

El algoritmo comienza generando una población de soluciones mediante la ejecución del *Método de Generación de la Diversidad*, que son luego mejoradas por el *Método de Mejora*. El conjunto de referencia se genera mediante la selección de las b ($b = b_1 + b_2$) mejores soluciones de la población. En primer lugar, se seleccionan las b_1 soluciones de la población con mejores valores de la función objetivo. Seguidamente, se seleccionan las b_2

```
procedimiento Búsqueda Dispersa
comienzo
   Generación de la Población;
   Generación del Conjunto de Referencia;
   repetir
    repetir
      Método de Generación de Subconjuntos;
      Método de Combinación de Soluciones;
      Método de Mejora;
    hasta (CriterioParada1);
    Método de Actualización del Conjunto de
    Referencia;
   hasta (CriterioParada2);
final.
```

Figura 1: Búsqueda Dispersa

soluciones de la población más dispersas con respecto a las soluciones incluidas en el conjunto de referencia. Una vez construido el conjunto de referencia, se generan diversos subconjuntos de soluciones del mismo que serán utilizados para la ejecución del *Método de Combinación*. Entonces se combinan las soluciones de cada uno de los subconjuntos anteriormente generados para obtener nuevas soluciones que se mejoran mediante el *Método de Mejora*. Finalmente, el *Método de Actualización del Conjunto de Referencia* actualiza el conjunto de referencia usando las soluciones mejoradas.

4. Aplicación de la Búsqueda Dispersa

El objetivo de esta sección es describir los procedimientos de la Búsqueda Dispersa propuestos para la resolución del problema de selección de instancias en Minería de Datos. Como se ha señalado anteriormente, en una Búsqueda Dispersa, un conjunto de soluciones del problema evoluciona por la aplicación de mecanismos guiados de diversificación e intensificación. La intensificación propuesta para la selección de instancias se implementa en el Método de Mejora diseñado para este problema. Consiste en añadir a la solución que se está mejorando aquellas instancias que han pertenecido a soluciones que han presentado

un buen comportamiento en el pasado. Para ello, se asigna un peso a cada instancia proporcional al poder de clasificación de las soluciones a las que ha pertenecido. Estos pesos se calculan en la fase inicial de generación del conjunto de referencia y se actualizan cada vez que este conjunto es actualizado. Por otra parte, la función objetivo que guía el proceso de búsqueda es el bien conocido porcentaje de acierto.

4.1. Generación de la Población

La población inicial está formada por 100 soluciones generadas siguiendo el procedimiento aleatorio que se explica a continuación. Se desordena de forma aleatoria el conjunto de instancias de entrenamiento, T, y se selecciona una única instancia de cada clase siguiendo el orden de instancias obtenido tras la reordenación aleatoria de T.

Dada una solución obtenida aleatoriamente $S = \{t_1, t_2, ...t_S\}$ y el conjunto de entrenamiento T, el valor de la función objetivo de dicha solución se calcula como

$$f_T(S) = 100\frac{|t \in T \setminus S : \tilde{c_t} = c_t|}{|T \setminus S|} \qquad (1)$$

donde c_t es la clase cierta de la instancia t y $\tilde{c_t}$ es la clase asignada a t por el clasificador del vecino más cercano.

4.2. Generación del Conjunto de Referencia

La generación del conjunto de referencia $ConjRef$ implica, en primer lugar, la selección de las 5 mejores soluciones de la población atendiendo a la calidad del valor de la función objetivo mostrada anteriormente. Para cada una de las 5 soluciones restantes que formarán parte del conjunto de referencia, se selecciona la solución de la población más dispersa con respecto a las soluciones ya incluidas en el conjunto de referencia. De esta forma, las soluciones del conjunto de referencia son soluciones buenas atendiendo tanto al criterio de calidad del valor de la función objetivo como al criterio de diversidad.

Sean dos soluciones al problema de selección de instancias, $S = \{t_1, t_2, ..., t_S\}$ y $S' = \{r_1, r_2, ..., r_{S'}\}$, formadas por el subconjunto de instancias seleccionadas, y sea I el conjuntos de instancias que pertenecen a alguna de las soluciones ya incluidas en el conjunto de referencia. Esto es,

$$I = \bigcup_{S \in ConjRef} S.$$

La diversidad, $Div(S)$, de una solución S con respecto al conjunto de instancias I viene dada por la diferencia simétrica entre S y I, $Div(S, I)$. Es decir,

$$Div(S) = Div(S, I) = |(S \cup I) \setminus (S \cap I)|.$$

El algoritmo propuesto para generar el conjunto de referencia se describe en la Figura 2.

1. Inicializar:
   - *a)* Sea $ConjRef$ un conjunto vacío.
   - *b)* Añadir a $ConjRef$ las b_1 mejores soluciones de la población.
   - *c)* Obtener el conjunto inicial de instancias: $I = \cup_{S \in ConjRef} S$.
   - *d)* $b = b_1$
2. Repetir:
   - *a)* Para cada $S \notin ConjRef$, calcular $Div(S, I)$.
   - *b)* Sea $S^* = \arg\max\{Div(S, I) : S \notin ConjRef\}$.
   - *c)* $ConjRef \leftarrow ConjRef \cup S^*$.
   - *d)* $b \leftarrow b + 1$.
   - *e)* Actualizar I.

   mientras $b < b_1 + b_2$.

Figura 2: Método de Generación del Conjunto de Referencia

Tras obtener el conjunto de referencia, para cada instancia, t, que aparezca en al menos

una solución de dicho conjunto, se calcula el siguiente valor

$$w_t = \sum_{t \in S, S \in ConjRef} \frac{1}{Error(S)} \qquad (2)$$

donde $Error(S)$ es una medida del error que se produce al clasificar usando la solución S. Es decir

$$Error(S) = 1 - \frac{f_T(S)}{100}.$$

con $f_T(S)$ dado por la ecuación (1).

De esta forma, cuanto mayor sean los errores de clasificación de las soluciones en las que aparece una determinada instancia, menor será su peso. A las instancias que no aparecen en el conjunto de referencia se les asigna un peso igual a cero.

4.3. Generación de Subconjuntos

Consideramos, como en las implementaciones básicas de la búsqueda dispersa, todos los subconjuntos de dos soluciones del conjunto de referencia. Las soluciones de los subconjuntos son entonces combinadas para construir nuevas soluciones.

4.4. Método de Combinación de Soluciones

Dadas dos soluciones, $S = \{t_1, t_2, ..., t_S\}$ y $S' = \{r_1, r_2, ..., r_{S'}\}$, pertenecientes a uno de los subconjuntos generados, se obtiene la nueva solución, S'', al intersecar las instancias de las soluciones a combinar. De esta forma se tiene

$$S'' = S \cap S'$$

Al combinar las soluciones S y S' para obtener la solución S'', se calcula además el valor m, que hace referencia al número medio de instancias de las soluciones combinadas. Es decir

$$m = \frac{|S| + |S'|}{2}$$

4.5. Método de Mejora

La solución S'' obtenida de la combinación de dos soluciones se mejora mediante la adición de instancias del conjunto de entrenamiento no incluidas en la solución. Para ello, se ordenan dichas instancias según el orden decreciente de sus pesos, y se seleccionan de forma ordenada las instancias con mayores pesos hasta que el cardinal de S'' sea igual a m. Así, si

$$w_{t(1)} \geq w_{t(2)} \geq w_{t(3)} \geq ...w_{t(|T|-|S''|)}$$

entonces

$$S'' = S'' \cup \{w_{t(1)}, w_{t(2)}, ..., w_{t(m-|S''|)}\}$$

4.6. Método de Actualización del Conjunto de Referencia

En este trabajo implementamos una actualización estática del conjunto de referencia. Es decir, se realizan todas las combinaciones y mejoras posibles obteniendo un nuevo conjunto de soluciones. A continuación, se seleccionan las b soluciones con mejor valor de la función objetivo entre las soluciones del nuevo conjunto y del conjunto de referencia.

5. Experiencia Computacional

Para evaluar el comportamiento de la Búsqueda Dispersa como método de selección de instancias se aplicó ésta a una serie de bases de datos de reducido tamaño. Las bases utilizadas están disponibles en el repositorio UCI (www.ics.uci.edu/~ mlearn/MLRepository.html). En este estudio preliminar se usaron las bases de datos Iris (150 instancias, 4 características, 3 clases), Glass (214 instancias, 9 características, 6 clases), Wine (178 instancias, 13 características, 3 clases), Led7 (500 instancias, 7 características, 10 clases) y Led24 (200 instancias, 24 características, 10 clases).

Los experimentos se realizaron siguiendo la metodología desarrollada en [4]. Así, para cada base de datos, se tomó la partición de 10 conjuntos que está disponible en (sci2s.ugr.es/keel/index.php) y se aplicó la Búsqueda Dispersa a los diez pares en los que un conjunto de esta partición actúa como conjunto de validación y el resto como conjunto de entrenamiento. En cada ejecución se anotó el porcentaje de reducción y el porcentaje de acierto en el conjunto de entrenamiento y de

validación. Los cuadros 1, 2, 3, 4 y 5 muestran los resultados obtenidos. En la primera columna se encuentran los algoritmos empleados en la comparativa: Generational Genetic Algorithm (CGA), Steady-Sate Genetic Algorithm (SGA), CHS Adaptive Search Algorithm, Population-Based Incremental Learning (PBIL) y Búsqueda Dispersa (BD). Las siguientes dos columnas dan el porcentaje de reducción promedio en el número de instancias y la desviación típica de estos valores. A continuación se muestran los porcentajes de acierto medios y sus desviaciones en los conjuntos de entrenamiento y validación, respectivamente. Los valores para los métodos CGA, SGA, PBIL y CHC se han obtenido de (sci2s.ugr.es/keel/index.php).

De los resultados obtenidos se concluye que la Búsqueda Dispersa es un método apropiado para el problema de la reducción de instancias. Los resultados obtenidos por la Búsqueda Dispersa son comparables, en general, a los obtenidos por los cuatro métodos evolutivos usados en la comparativa. Con respecto al método evolutivo CHC destacado en [4] como es el más apropiado de los cuatro, la búsqueda dispersa obtiene ligeramente peores reducciones en el conjunto de instancias, pero para algunas de las bases de datos, le supera en el porcentaje de acierto en el conjunto de entrenamiento e incluso, lo que es más importante, en el conjunto de validación o test. En comparación con los otros tres métodos evolutivos, los resultados de la búsqueda dispersa son superiores o equivalente en la mayoría de los casos.

6. Conclusiones

Hemos diseñado e implementado una Búsqueda Dispersa para el problema de la Selección de Instancias. Además, hemos comparado los resultados obtenidos con la Búsqueda Dispersa con los obtenidos con otros algoritmos evolutivos previamente propuestos. Para ello hemos usado un conjunto de cinco bases de datos. Aunque la experiencia computacional no ha sido extensa, los resultados obtenidos son prometedores. Como líneas futuras de investigación pretendemos mejorar parte del diseño de nuestra Búsqueda Dispersa así como ampliar la experiencia computacional mediante el uso de bases de datos de mayor tamaño.

Referencias

[1] D. W. Aha and D. Kibler and M. K. Albert, *Instance-Based Learning Algorithms*, Machine Learning, vol. 6, pp. 37-66,1997.

[2] H. Brighton and C. Mellish, *Advances in Instance Selection for Instance-Based Learning Algorithms*, Data Mining and Knowledge Discovery, vol. 6, pp. 153-172,2002.

[3] C. E. Broadley, *Addresing the Selective Superiority Problem: Automatic Algorithm/Model Class Selection*, in Proc. 10th International Machine Learning Conference, pp. 17-24,1993.

[4] J. R. Cano and F. Herrera and M. Lozano, *Using Evolutionary Algorithms as Instance Selection for Data Reduction in KDD: An Experimental Study*, Intitute of Electrical and Electronics Engineers Transactions on Evolutionary Computation, vol. 7, No. 6,2003.

[5] T. M. Cover and P. M. Hart, *Nearest Neighbor Pattern Classification*, Intitute of Electrical and Electronics Engineers Transactions on Information Theory, vol. 10, pp. 57-78,1967.

[6] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice Hall international, Englewood Cliffsin, 1982.

[7] Glover, F., Heuristics for Integer Programming using Surrogate Constraints, *Decision Sciences* 8, (1977) 156–166

[8] Glover, F., Laguna, M., Martí, R. Fundamentals of Scatter Search and Path Relinking *Control and Cybernetics*, 39, (2000) 653-684

Alg.	Reducción		1NN			
			Acierto Entr.		Acierto Val.	
		DT	Media	DT	Media	DT
GGA	85,93 %	1,19 %	78,82 %	1,76 %	67,96 %	7,00 %
SGA	86,45 %	1,12 %	80,07 %	1,28 %	68,72 %	9,12 %
PBIL	91,22 %	0,89 %	77,88 %	1,56 %	67,62 %	6,46 %
CHC	92,99 %	0,87 %	76,79 %	2,11 %	59,19 %	7,76 %
BD	90.29 %	5.23 %	78,33 %	3,00 %	61,94 %	9,77 %

Cuadro 1: Glass

Alg.	Reducción		1NN			
			Acierto Entr.		Acierto Val.	
		DT	Media	DT	Media	DT
GGA	83,44 %	1,28 %	58,32 %	1,76 %	34,69 %	7,18 %
SGA	83,67 %	1,24 %	59,61 %	1,27 %	37,96 %	8,73 %
PBIL	87,11 %	1,58 %	57,39 %	1,53 %	39,51 %	7,06 %
CHC	91,55 %	0,68 %	55,00 %	0,61 %	25,66 %	8,34 %
BD	93.61 %	1,56 %	68.50 %	10,01 %	69,99 %	8,66 %

Cuadro 2: Led24

Alg.	Reducción		1NN			
			Acierto Entr.		Acierto Val.	
		DT	Media	DT	Media	DT
GGA	89,58 %	0,76 %	44,34 %	2,91 %	45,13 %	4,04 %
SGA	93,69 %	0,53 %	45,23 %	4,44 %	42,63 %	3,46 %
PBIL	95,64 %	0,51 %	36,31 %	1,08 %	33,40 %	4,31 %
CHC	96,93 %	0,24 %	76,31 %	0,27 %	61,13 %	5,66 %
BD	95,84 %	2,65 %	73,07 %	2,02 %	70,60 %	6,00 %

Cuadro 3: Led7

Alg.	Reducción		1NN			
			Acierto Entr.		Acierto Val.	
		DT	Media	DT	Media	DT
GGA	93,63 %	1,02 %	99,19 %	0,51 %	95,00 %	3,00 %
SGA	95,00 %	0,51 %	99,31 %	0,57 %	97,19 %	2,81 %
PBIL	94,94 %	0,95 %	79,28 %	1,62 %	71,96 %	11,12 %
CHC	96,82 %	0,46 %	80,15 %	1,97 %	65,78 %	6,35 %
BD	89,45 %	1,04 %	95,45 %	4,27 %	94,90 %	4,70 %

Cuadro 4: wine

Alg.	Reducción		1NN			
			Acierto Entr.		Acierto Val.	
		DT	Media	DT	Media	DT
GGA	94,81 %	0,74 %	98,37 %	0,65 %	94,00 %	4,80 %
SGA	95,11 %	0,53 %	98,44 %	0,40 %	95,33 %	5,60 %
PBIL	96,52 %	0,61 %	97,70 %	0,81 %	98,00 %	3,20 %
CHC	96,30 %	0,59 %	98,22 %	0,65 %	93,33 %	8,00 %
BD	95,41 %	3,09 %	97,55 %	5,78 %	95,33 %	4,26 %

Cuadro 5: Iris

[9] P. E. Hart, *The Condensed Nearest Neighbor Rule*, Intitute of Electrical and Electronics Engineers Transactions on Information Theory, vol. IT-14, pp. 515-516, 1968.

[10] D. kibbler and D. W. Aha, *Learning Representative Exemplars of Concepts: An Initial Case of Study*, in Proc. 4th International Workshop Machine Learning, pp. 24-30, 1987.

[11] Laguna, M. and R. Martí, *Scatter Search: Metodology and Implementations in C*, Kluwer Academic Press, (2003).

[12] D. G. Lowe, *Similarity Metric Learning for a Variable-Kernel Classifier*, Neural Computation, vol. 7, no. 1, pp. 72-85, 1995.

[13] R. Martí, Laguna, M. *Scatter Search: diseño básico y estrategias avanzadas*, Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial. Numero 19, Volumen 2, (2003) páginas 123-130.

[14] D. B. Skalak, *Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms*, in Proc. 11th International Conference Machine Learning, pp. 72-85, 1994.

[15] D. L. Wilson, *Asymtotic Properties of Nearest Neighbor Rules using Edited Data*, Intitute of Electrical and Electronics Engineers Transactions Systems Man, vol. SMC-2, pp. 408-421, 1972.

[16] D. R. Wilson and T. R. Martinez, *Instance Pruning techniques*, in Proc. 14th International Conference Machine Learning, pp. 403-411, 1997.

[17] D. R. Wilson and T. R. Martinez, *Improved Heterogeneous Distance Functions*, Journal of Artificial Intelligence Research, vol. 6, No. 1 pp. 1-34,1997.

[18] D. R. Wilson and T. R. Martinez, *Reduction Techniques for Instance-Based Learning Algorithms*, Machine Learning, vol. 38, pp. 257-268,1997.