Edición de muestras basada en búsqueda secuencial

Olvera López J. Arturo
Carrasco Ochoa J. Ariel
Martínez Trinidad J. Francisco

Reporte Técnico No. CCC-05-002
7 de Febrero de 2005

Luis Enrique Erro 1
Sta. Ma. Tonantzintla,
72840, Puebla, México.

Edición de muestras basada en búsqueda secuencial

Olvera López J. Arturo
Carrasco Ochoa J. Ariel
Martínez Trinidad J. Francisco

Coordinación de Ciencias Computacionales,
Instituto Nacional de Astrofísica, Óptica y Electrónica,
Luis Enrique Erro 1, Sta. Ma. Tonantzintla,
72840, Puebla, México
{aolvera,ariel,fmartine}@inaoep.mx

Resumen. Los clasificadores supervisados basan su aprendizaje en un conjunto de datos denominado conjunto de entrenamiento, mediante el cual, se proporciona al clasificador una serie de casos o situaciones con las que puede encontrarse al requerirse una predicción o clasificación de un nuevo objeto.
Debido a que en muchas ocasiones, no es útil en su totalidad la información que proporciona la muestra o conjunto de entrenamiento, es necesario editar o preprocesar tal muestra de modo que se considere sólo aquellos objetos relevantes para fines de clasificación. Dentro del campo de la edición de muestras se han propuesto diversos métodos que ofrecen una solución a dicho problema, en este trabajo se propone un método de edición de muestras basado en búsqueda secuencial, se compara contra algunos de los métodos tradicionales y se presentan algunos resultados experimentales.

Palabras clave. Edición de muestras, clasificación, búsqueda secuencial, selección de objetos.

1. Introducción

En la clasificación supervisada se requiere de un conjunto de información que se proporciona a los clasificadores, a tal conjunto se le denomina conjunto de entrenamiento, el cual denotaremos con *T*. Dicho conjunto es la base de los clasificadores supervisados para proporcionar una predicción de los nuevos objetos que se presenten. Cuando los clasificadores emplean en cada momento a todo *T* para clasificar a un objeto, se les denomina clasificadores basados en instancias (*IBC*).

Cuando un nuevo objeto *O* se presenta para ser clasificado, el objetivo del clasificador es determinar (a partir de la información proporcionada por *T*) la clase o etiqueta que se asignará a *O*. Para cada nuevo objeto, un *IBC* realiza los cálculos de similitud entre éste y todos los objetos de *T*, si se tiene una gran cantidad de objetos, el tiempo empleado por el clasificador puede ser muy grande. Por otra parte, no se garantiza que todos los elementos de *T* sean útiles o proporcionen información relevante para el proceso de clasificación, ya que suelen presentarse elementos superfluos para tal proceso, que pueden considerarse como objetos ruidosos o redundantes. Por todo esto es necesario aplicar al conjunto *T* un método de edición, es decir, obtener un nuevo conjunto *S* a partir de *T* ($S \subseteq T$) tal que $|S|<|T|$ y *S* contenga los objetos con los cuales el clasificador utilizado obtenga porcentajes de clasificación correcta mayores que los obtenidos con *T*.

De acuerdo a [1], el proceso de edición puede llevarse acabo siguiendo tres distintas estrategias o direcciones: *Incremental, decremental* y *por lotes.*

Estrategia incremental. En este tipo de estrategia (figura 1) se parte de un conjunto vacío *S* y en cada paso se añade a *S* el objeto que satisface el criterio de selección de objetos empleado. En esta estrategia, el orden en que se presentan los objetos en el conjunto es importante, ya que la

probabilidad de que los primeros objetos sean incluidos en S es mayor que la de los últimos. Es decir, cuando los últimos objetos se presentan pueden ya estar representados por algunos de los primeros. En este sentido, puede verse dañada la precisión en la clasificación si los últimos objetos representan una mayor generalización que los primeros. Es por esta razón que en la estrategia incremental los objetos se presentan en un orden aleatorio, ya que por definición, un método incremental debe ser capaz de seleccionar objetos sin necesidad de que éstos se presenten primero.

Una ventaja de este tipo de estrategia es que resulta ser más rápida y consumir menos recursos de almacenamiento durante el proceso de entrenamiento del clasificador en comparación a las estrategias no incrementales.

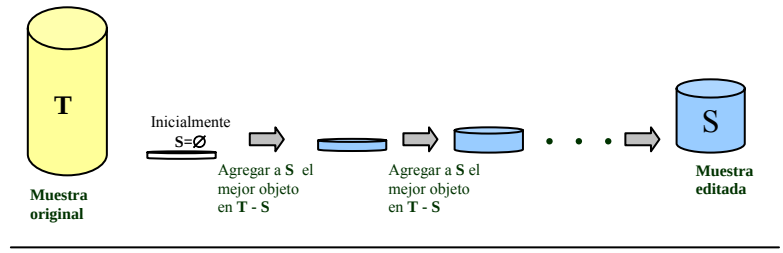

Figura 1. Estrategia de edición incremental

La principal desventaja de la estrategia incremental es, como se ha mencionado anteriormente, que es sensible al orden en que se presentan los objetos y además las primeras decisiones están basadas en muy poca información y por tanto estas decisiones son propensas a errar en la clasificación. Por esta razón, algunos métodos incrementales realizan una fase denominada de grupo inicial, que consiste en partir de un determinado número de objetos en el conjunto S y después aplicar la estrategia.

Estrategia decremental. Esta estrategia comienza con $S=T$ y en cada paso se determina el objeto a eliminar de S de acuerdo al criterio de selección de objetos (figura 2). También en esta estrategia es importante el orden en que los objetos se presentan, pero a diferencia de las técnicas incrementales todos los objetos parcialmente almacenados están disponibles en todo momento para examinar cuál de ellos resulta conveniente eliminar.

La principal ventaja en este tipo de estrategia es que se obtiene una mayor reducción de T y normalmente con el subconjunto S que se obtiene, se logra una mayor precisión de clasificación con respecto a la obtenida con la muestra original.

Una desventaja que presenta esta estrategia es que resulta ser computacionalmente más costosa con respecto al enfoque incremental, ya que, por ejemplo, para encontrar similitud entre un objeto y el subconjunto S, la estrategia decremental lleva a acabo n comparaciones (donde $n=|S|$), mientras que la estrategia incremental realiza menos cálculos (cero inicialmente y posteriormente sólo una fracción de $|T|$).

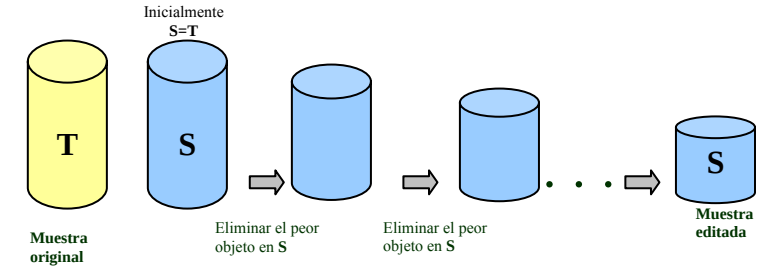

Figura 2. Estrategia de edición decremental

Estrategia por lotes. Esta es otra de las maneras en que puede llevarse a cabo el proceso de edición**,** la cual consiste en identificar y marcar aquellos objetos que no satisfacen el criterio de selección, los cuales no serán considerados en el subconjunto S y finalmente se eliminan tales objetos, es decir, no se elimina sólo un objeto sino grupos de estos. Al igual que la estrategia decremental, esta técnica resulta ser computacionalmente costosa.

También, en base al efecto que causa la eliminación de los objetos, suele dividirse a los métodos de edición en tres esquemas [2]: *Incremento de la competencia, Preservación de la competencia* y *Esquema híbrido.*

Incremento de la competencia. Este esquema se enfoca a eliminar aquellos objetos, los cuales, al ser descartados, la precisión en los resultados de clasificación se incrementa. Normalmente, esta técnica elimina objetos considerados como ruido.

Preservación de la competencia. Esta técnica elimina objetos superfluos, es decir, aquellos objetos cuya eliminación no provoca un decremento en la precisión de los resultados de clasificación.

Esquema híbrido. Se deriva de los dos esquemas anteriores y se encarga de abordar ambos problemas a la vez.

En este reporte se presenta un método de edición, el cual corresponde a la estrategia decremental y al esquema hibrido. En la sección 2 se presentan algunos de los trabajos relacionados con la edición de muestras, la sección 3 presenta un método propuesto basado en las ideas de la búsqueda secuencial y finalmente en la sección 4 se presentan los resultados preliminares obtenidos al aplicar el método propuesto.

2. Trabajo relacionado

Se han realizado diversos trabajos para llevar a cabo la edición de muestras siguiendo cada una de las estrategias, a continuación se describen brevemente algunos de estos trabajos.

Hart [3] propone la regla del vecino más cercano condensado (*CNN*). Este método incremental consiste en encontrar de entre los elementos de T (muestra original) a un subconjunto S tal que cada objeto de T sea más cercano o parecido a los objetos de S con la misma clase que a los que tienen distinta clase. Este subconjunto S es utilizado para clasificar correctamente todos los objetos en T. Además se asume que el conjunto T es consistente, es decir, que dentro de éste no existen dos objetos cuyos atributos sean idénticos y correspondan a clases distintas. Este método comienza seleccionando de manera aleatoria un objeto correspondiente a cada una de las distintas clases y estos objetos se añaden a S, el cual inicialmente es un conjunto vacío. Posteriormente, cada objeto en T es clasificado empleando únicamente los objetos de S; cuando un objeto es clasificado erróneamente entonces éste se añade a S, para garantizar que será clasificado correctamente. El proceso se repite hasta que no existan objetos en T que sean clasificados de manera errónea. El método *CNN* se muestra en la figura 3.

CNN(Training set T): Object set S
$S = \varnothing$
Repeat
  Additions= FALSE
  For all patterns in T do
    Randomly pick O from T
    Find $s_c \in S$ such that $Distance(O,s_c) = \min_j Distance(O,s_j)$
    If class(O) $\neq$ class(s_c) then
      $S = S \cup \{O\}$
      Additions = TRUE
Until NOT(Additions)
Return S

Figura 3. Método *CNN*

Esta técnica es sensible al ruido, ya que objetos ruidosos suelen ser clasificados erróneamente por sus vecinos y de este manera, los objetos ruidosos se anexan a *S*, lo cual provoca dos inconvenientes: el primero es que no se logra una reducción considerable de la muestra, ya que los objetos ruidosos son innecesarios pero aún siguen presentes y el segundo inconveniente es el efecto negativo que el subconjunto resultante causa en los resultados de clasificación, debido a que los objetos ruidosos no aportan información relevante al clasificador.

Ritter et al. [4], proponen la regla selectiva del vecino más cercano (*SNN*), la cual es una extensión de *CNN*. De a cuerdo a *SNN,* los objetos de *T* estarán más cerca a objetos de *S* con la misma clase que a los objetos de *T* con distinta clase, además, *SNN* garantiza encontrar un conjunto mínimo que satisface tal condición.

Gates [5] realiza una extensión decremental de *CNN* con la regla del vecino más cercano reducido (*RNN*), la cual comienza con $S=S_{CNN}$, donde S_{CNN} es el conjunto de objetos obtenido con *CNN* a partir de *T*. Posteriormente, se elimina un objeto de *S* si tal eliminación no causa que algún otro objeto de *T* sea clasificado erróneamente por los objetos restantes de *S*. Con esta regla se obtiene un subconjunto de *CNN*, por lo que el conjunto editado, en cuanto al número de objetos, es menor con respecto al obtenido con *CNN*. La regla *RNN* se muestra en la figura 4.

RNN(Training set T, *CNN* set S_{CNN}) : object set *S*

1. $S=S_{CNN}$
2. Remove the first object from *S*
3. Use *S* to classify all objects in *T*:
   a) If all objects are classified correctly, go to 4
   b) If an object is classified incorrectly, return the object that was removed and go to 4
4. If every object in *S* has been removed once (and posibly replaced) then halt. Otherwise, remove the next object and go to 3.

Return *S*

Figura 4. Método *RNN*

Wilson [6] presenta la regla del vecino más cercano editado (*ENN*), la cual se muestra en la figura 5 y cuyo criterio de selección de objetos es el siguiente: "*Eliminar o descartar el objeto en cuestión si la clase de éste no coincide con la de la mayoría de sus k vecinos más cercanos (k-NN)"*.

ENN(Training set T): object set *S*

1. $S=T$
2. For each object O_i in *S do*
   a) Find the k nearest neighbors of O_i in S – $\{O_i\}$
   b) Remove O_i from *S* if its label disagrees with the class associated with the largest number of *k* nearest neighbors
3. Return *S*

Figura 5. Método *ENN*

Esta técnica decremental suele emplearse para filtrar de ruido a una muestra, ya que, se elimina a aquellos objetos raros (ruidosos) en vecindarios cuyos objetos corresponden a la misma clase. Una pequeña modificación es presentada en [11] mediante la cual se sigue la idea de *ENN* pero se sustituye a *k-NN* por *k-NCN (k Nearest Centroid Neighbourhood)*.

Una variante de *ENN* es *RENN* (*Repeated ENN*) [7], que consiste en aplicar el algoritmo *ENN* de manera repetida hasta que todos los objetos en *S* tengan la misma clase que la mayoritaria de sus *k* vecinos más cercanos.

Una extensión de *ENN* fue realizada por Tomek [7] con el método de edición por lotes denominado *All k-NN* (figura 6) que consiste en que para cada objeto en *S*, se determina cómo

lo clasifican sus i vecinos más cercanos (i = 1 .. k, con k = número de vecinos), si éstos lo clasifican erróneamente, entonces al objeto se le asigna una marca, una vez que se han analizado todos los objetos de la muestra, se eliminan los objetos que resultaron marcados.

All *k*-NN((Training set T, number of neighbors *k*): object set *S*

1. *S*=*T*
2. For each object *O* in *S* do
   a) *i*=1, flag(*O*)=1
   b) find *i* nearest neighbors of *O* : NN(*i*,*O*)
   c) If the majority of NN(*i*,*O*) classify *O* incorrectly, flag(*O*)=0
   d) *i* = *i* + 1
   e) If $i \leq k$ go to step b)
3. Eliminate from *S* those objects with flag(*O*)=0
4. Return *S*

Figura 6. Método *All k-NN*

Aha et al. [8] y [9] proponen una serie de métodos de edición incremental denominados *IB2, IB3, IB4* e *IB5*. Estos métodos emplean como base el algoritmo *IB1*(figura 7), el cual no es un método de edición, ya que es una técnica para determinar si la clasificación de un nuevo objeto *O* es correcta o incorrecta. La manera en que se determina la clasificación es encontrando en el conjunto de entrenamiento al objeto más parecido a *O* y si sus correspondientes clases difieren, entonces la clasificación de *O* es incorrecta.

IB1((Training set T)
CD←∅
For each object *O* in *T* do
1. For each *y* in *CD* do
   Sim[y]←Similarity(*O*,*y*)
2. y_{max} ←some $y \in CD$ with maximal Sim[*y*]
3. If class(*O*) = class(y_{max})
   Then classification ← correct
   Else classification ← incorrect
4. $CD \leftarrow CD \cup \{O\}$

Figura 7. Método *IB1*

El algoritmo *IB2* es idéntico a *IB1,* excepto que *IB2* almacena los objetos clasificados erróneamente, pues es un método de edición cuya regla a seguir es encontrar en la muestra original un subconjunto que contenga aquellos objetos que fueron clasificados incorrectamente durante el proceso. *IB2* resulta ser sensible al ruido, pues en base a la regla que sigue, almacena objetos ruidosos, ya que, por su naturaleza, este tipo de objetos suelen clasificarse de manera incorrecta. *IB3* es una extensión de *IB2*, en la cual básicamente se evita almacenar todos los objetos ruidosos y considerar de entre ellos a aquellos que no afectarán en gran parte los resultados de clasificación. *IB3* analiza los resultados de clasificación antes de eliminar un objeto ruidoso, mantiene un registro de cómo se clasifica con los objetos que se van almacenando y elimina aquellos con los cuales, estadísticamente se ven afectados los resultados de clasificación. *IB4* e *IB5* son extensiones de *IB3,* ya que, para cada clase determinan un conjunto de pesos que serán asignados a los atributos de los objetos para fines de cálculo de similitudes.

Wilson y Martínez [1] proponen los métodos decrementales *DROP (Decremental Reduction Optimization Procedure)*. Estos métodos basan su regla de selección en términos del concepto de socio. El socio de un objeto *O* es aquél objeto que tiene a *O* como uno de sus *k* vecinos más cercanos. *DROP1* (figura 8) elimina un objeto *O* de *S* si sus socios en *S* se clasifican correctamente sin *O*, es decir, bajo este criterio, la ausencia de *O* no afecta en la clasificación.

DROP2 verifica el efecto que causa la eliminación del objeto en los objetos de la muestra original *T*, es decir, *DROP2* elimina al objeto *O* de *S* si los socios que *O* tiene en *T* clasifican

correctamente sin *O*. *DROP3* y *DROP4* aplican un filtrado de ruido (similar al *ENN*) antes de comenzar el proceso de edición. La diferencia entre ambos es el criterio empleado en la etapa de filtrado, ya que *DROP4* antes de eliminar el objeto ruidoso, verifica el impacto de clasificación provocado al no considerar tal objeto para determinar si será o no eliminado. Finalmente, el método *DROP5* modifica *DROP2* de tal manea que comienza por eliminar objetos que se encuentran cerca de los enemigos más cercanos (objetos cercanos con distinta clase).

```
DROP1( Training set T ): Object set S
   Let S=T
   For each object O in S:
    Find O.N1..k+1, the k+1 nearest neighbors of O in S
    Add O to each of its neighbors' lists of associates
   For each object O in S:
        Let with = # of associates of O classified correctly with O as a neighbor.
        Let without = # of associates of O classified correctly without O.
        If (without - with) ≥ 0
            Remove O from S
            For each associate A of O
                  Remove O from A's list of nearest neighbors
                   Find a new nearest neighbor for O.
                   Add A to its new nearest neighbor's list of associates
            For each neighbor N of O
                   Remove O from N's lists of associates
        Endif
   Return S
```

Figura 8. Método *DROP1*

Otro método de edición es *ICF (Iterative Case Filtering)* [2], cuya regla de selección por lotes se basa en los conjuntos *alcance(O)* y *cobertura(O)* del objeto *O*, los cuales, de manera análoga, se refieren al vecindario y conjunto de socios respectivamente. La regla de selección es la siguiente: eliminar aquellos objetos tales que el tamaño de *alcance* es mayor que el de *cobertura*, lo cual quiere decir que un objeto *O* será eliminado cuando mediante otros objetos se generaliza la información que pudiera proporcionar. Como etapa inicial, *ICF* filtra la muestra empleando *ENN*. El método *ICF* se muestra en la figura 9.

```
ICF ( Training set T )
// Perform Wilson Editing
For all O∈ T do
  If O classified incorrectly by k nearest neighbors then
    Flag O for removal
For all O∈ T do
  If O flagged for removal then T = T – {O}
Repeat
  For all O ∈ T do
    Compute recheable(O)
    Compute coverage(O)
  Progress = false
  For all O ∈ T do
    If |recheable(O)| > |coverage(O)| then
      Flag O for removal
      Progress = True
   For all O ∈ T do
     If O flagged for removal then T = T – {O}
 Until not Progress
Return T
```

Figura 9. Método *ICF*

Riquelme et al. [10] realizan edición con proyecciones ordenadas, tal método decremental se basa en el concepto de *debilidad(O)*, lo cual se refiere (en términos gráficos de representación espacial de los objetos) al número de veces que el objeto *O* no es un borde o límite en una partición (se encuentra cerca de otra partición). La regla de edición consiste en eliminar aquellos objetos cuya *debilidad* es igual al número total de atributos.

Otra estrategia empleada en la edición consiste en re-etiquetar objetos, es decir, cambiar la clase a la que los objetos pertenecían originalmente, de tal manera que se mejore la precisión de la clasificación, por ejemplo, en [12] se usa un ensamble de redes neuronales y los elementos de *T* se re-etiquetan de acuerdo a la predicción del ensamble. Puede notarse que este tipo de edición se enfoca a incrementar los resultados de clasificación sin tener como objetivo reducir el tamaño de *T*.

Además de re-etiquetar, otras técnicas también eliminan objetos, Barandella y Gasca [13] presentan un algoritmo de depuración (basado en la edición generalizada [14]) mediante el cual, dependiendo del valor de dos parámetros *k* y *k'* el objeto es eliminado o re-etiquetado.

Una estrategia distinta a todas las descritas anteriormente se presenta en [15] que consiste en la obtención de *S* mediante la mezcla de objetos. La manera en que este método funciona consiste en encontrar dos objetos cercanos correspondientes a la misma clase y mezclarlos para obtener un nuevo objeto, tal proceso se repite hasta que los resultados de clasificación comienzan a sufrir un decremento.

Las estrategias de re-etiquetado y mezcla pueden ser empleadas como solución al problema de edición pero se debe estar consciente de que este tipo de estrategias, al modificar atributos o clases, alteran la naturaleza original del problema.

3. Estrategia propuesta

El método de edición que se presenta en este reporte consiste en adaptar la idea de la búsqueda secuencial hacia atrás (*BSS*) [16] para la selección de objetos. A este método de edición adaptado se le denominará *Backward Sequential Edition (BSE)*.

En un conjunto de entrenamiento *T,* suele ocurrir que algunos de los objetos de *T* no aportan información relevante para la clasificación, por lo que es necesario identificar y descartar tales objetos, es decir, realizar una edición o selección de objetos, lo cual es un problema de búsqueda que consiste en encontrar el subconjunto de objetos óptimo para el entrenamiento del clasificador.

Debido a que el espacio de subconjuntos de un total de *d* objetos es de tamaño 2^d, los algoritmos para la selección de objetos suelen evitar emplear las técnicas exhaustivas, es decir, aquellas con las cuales se analizan las 2^d posibilidades, lo que representa un alto costo computacional, ya que éste resulta ser exponencial ($O(2^d)$). Una de las técnicas no exhaustivas es la búsqueda secuencial, cuyo orden de complejidad es polinomial ($O(d^2)$).

El método propuesto *BSE* (figura 10) es una técnica decremental no exhaustiva para la selección de objetos, el cual en cada paso descarta o elimina el objeto que menos información aporta en la calidad del subconjunto parcial. Para evaluar los subconjuntos parciales a lo largo del proceso se emplea un clasificador. La función *Classifier(P)* regresa como resultado el porcentaje de clasificación correcta con dicho clasificador empleando a *P* como conjunto de entrenamiento. En los experimentos realizados en este reporte se utilizó como clasificador a *k-Nearest Neighbors (k-NN)* [17] con *k*=3.

```
BSE(Training set: T ): object set S
 Let S=T
 BestEval=Classifier(S)
 Repeat
   WorstO = None
   For each object O in S
      S'= S – {O}
      If Classifier(S')≥ BestEval
       Then WorstO = O
            BestEval = Classifier(S')
   If WorstO ≠ None
    Then S = S – {WorstO}
 Until WorstO == None or S == ∅
 Return S
```

Figura 10. Método *BSE*

Función de distancia

Dados dos objetos O_1, O_2, cada uno definido por un conjunto de atributos, una función de distancia es aquella métrica que el clasificador emplea para determinar el parecido existente entre O_1 y O_2 de acuerdo al valor de los atributos. También suele decirse que la función de distancia calcula la similitud entre dos objetos (función de similitud). Mientras más parecidos sean dos objetos, la distancia entre estos es menor y de manera contraria sucede para objetos lejanos.

Una función de distancia comúnmente empleada es la distancia euclidiana, la cual se define como:

$$d(O_1,O_2)=\sqrt{\sum_{i=1}^{n}(x_i(O_1)-x_i(O_2))^2} \qquad (1)$$

Donde O_1, O_2 son los objetos de los cuales se calculará la distancia, n es el número de atributos y $x_i(O_1)$, $x_i(O_2)$ es el valor de la variable x para el atributo i en los objetos O_1, O_2. Otras funciones empleadas para el cálculo de distancias se muestran en la figura 11.

La distancia euclidiana, al igual que las distancias mostradas en la figura 11 se emplea en el caso en que todos los atributos del objeto son valores numéricos.

Además de los atributos numéricos, existen los atributos cuyos valores son no numéricos, en los cuales el atributo puede tomar un valor de entre un conjunto finito, por ejemplo colores, razas, enfermedades, etc.

Minkowsky :

$$D(O_1,O_2)=\left(\sum_{i=1}^{n}\left|x_i(O_1)-x_i(O_2)\right|^r\right)^{1/r}$$

Manhattan :

$$D(O_1,O_2)=\sum_{i=1}^{n}\left|x_i(O_1)-x_i(O_2)\right|$$

Camberra :

$$D(O_1,O_2)=\sum_{i=1}^{n}\frac{\left|x_i(O_1)-x_i(O_2)\right|}{\left|x_i(O_1)+x_i(O_2)\right|}$$

Chebychev :

$$D(O_1,O_2)=\max_{i=1}^{n}\left|x_i(O_1)-x_i(O_2)\right|$$

Cuadrática :

$$D(O_1,O_2)=(O_1-O_2)^T Q(O_1-O_2)=\sum_{j=1}^{n}\left(\sum_{i=1}^{n}(x_i(O_1)-x_i(O_2))q_{ij}\right)(x_j(O_1)-x_j(O_2))$$

Q es una matriz de peso (positiva), de dimensiones nxn

Mahalanobis :

$$D(O_1,O_2)=[detV]^{1/n}(O_1-O_2)^T V^{-1}(O_1-O_2)$$

V es la matriz de covarianza de $A_1 \ldots A_n$. A_j es el vector de valores para el atributo j en el conjunto de entrenamiento

Correlación :

$$D(O_1,O_2)=\frac{\sum_{i=1}^{n}(x_i(O_1)-\overline{x_i(O_1)})(x_i(O_2)-\overline{x_i(O_2)})}{\sqrt{\sum_{i=1}^{n}(x_i(O_1)-\overline{x_i(O_1)})^2\sum_{i=1}^{n}(x_i(O_2)-\overline{x_i(O_2)})^2}}$$

$\overline{x_i(O_1)}=\overline{x_i(O_2)}$ es el valor promedio para el atributo i en el conjunto de entrenamiento

Chi - cuadrada

$$D(O_1,O_2)=\sum_{i=1}^{n}\frac{1}{sum_i}\left(\frac{x_i(O_1)}{size_{O_1}}-\frac{x_i(O_2)}{size_{O_2}}\right)^2$$

sum_i es la suma de todos los valores del atributo i en el conjunto de entrenamiento y $size_{O_1}$ es la suma de todos los valores de los atributos del objeto O_1

Correalción categórica de Kendall :

$$D(O_1,O_2)=1-\frac{2}{N(N-1)}\sum_{i=1}^{n}\sum_{j=1}^{i-1}sign(x_i(O_1)-x_j(O_1))sign(x_i(O_2)-x_j(O_2))$$

$$sign(y)=\begin{cases}-1 & \text{si } y<0\\ 0 & \text{si } y=0\\ 1 & \text{si } y>0\end{cases}$$

N es el número total de objetos en el conjunto de entrenamiento

---

Figura 11. Distintas funciones para el cálculo de distancia entre objetos

En la literatura, se han propuesto funciones para calcular la distancia entre atributos no numéricos, por ejemplo, una de estas funciones es *VDM (Value Difference Metric)* [18], con la cual, la similitud entre dos valores $x_i(O_1)$, $x_i(O_2)$ del atributo *i* con respecto a los objetos O_1, O_2 es:

$$vdm_i(x_i(O_1), x_i(O_2)) = \sum_{c=1}^{C} \left(\frac{N_{i,x_i(O_1),c}}{N_{i,x_i(O_1)}} - \frac{N_{i,x_i(O_2),c}}{N_{i,x_i(O_2)}} \right)^2 \quad (2)$$

Donde, $N_{i,x_i(O_1)}$ es el número de veces (en el conjunto de entrenamiento) que el atributo *i* tiene valor *x* , $N_{i,x_i(O_1),c}$ es el número de veces que el atributo *i* tiene valor *x* en la clase *c* y *C* es el número total de clases.

Es común que los clasificadores tengan que enfrentarse a problemas en los que los atributos de los objetos son heterogéneos, es decir, están descritos por ambos tipos de valores (numéricos y no numéricos), entonces es necesario emplear una función heterogénea de distancia, por ejemplo, *HVDM (Heterogeneous Value Difference Metric)* [1], mediante la cual es posible calcular distancias entre objetos cuyos atributos son heterogéneos. *HVDM* se define de la siguiente manera:

$$HVDM(O_1, O_2) = \sqrt{\sum_{i=1}^{n} d_i^2(x_i(O_1), x_i(O_2))} \quad (3)$$

Donde la función $d_i(x_i(O_1), x_i(O_2))$ es la distancia para el atributo *i* y se define como:

$$d_i(x_i(O_1), x_i(O_2)) = \begin{cases} 1 & \text{Si } x_i(O_1) \text{ ó } x_i(O_2) \text{ son atributos faltantes} \\ vdm_i(x_i(O_1), x_i(O_2)) & \text{Si } i \text{ es no numérico} \\ \dfrac{|x_i(O_1) - x_i(O_2)|}{4\sigma_i} & \text{Si } i \text{ es numérico} \end{cases} \quad (4)$$

$vdm_i(x_i(O_1), x_i(O_2))$ es la función de la ecuación 2 y σ_i es la desviación estándar correspondiente al atributo *i* (en el conjunto de entrenamiento). La función *HVDM* se utilizó en los experimentos realizados en este reporte. Otras funciones de distancia que pudieran utilizarse se encuentran en [19] y [20].

4. Resultados experimentales y discusión

En esta sección se presentan los resultados obtenidos al emplear el método de edición decremental *BSE* descrito en la sección 3 y otros tres métodos decrementales. Además se discuten y analizan de manera comparativa los resultados obtenidos.

Para llevar a cabo los experimentos se utilizaron conjuntos o bases de datos obtenidos del repositorio Machine Learning Database de la universidad de California, Irvine [21]. De las bases de datos disponibles en el repositorio, para esta fase experimental, se eligieron 25 bases de datos, cuya descripción se muestra en la tabla 1.

Debido a que el método *BSE* sigue la estrategia decremental, se consideró apropiado, para fines de análisis y comparación de resultados, aplicar otros métodos decrementales además de *BSE*.

En [1] se realizaron experimentos de edición entre métodos incrementales y decrementales y en base a los resultados, los métodos decrementales *DROP, ENN* y *RENN* obtuvieron mejores resultados de clasificación con respecto a los métodos incrementales, por lo que se consideró interesante comparar estos métodos contra *BSE*.

Base de datos	Objetos	Atributos	Clases
Breast cancer (WI)	699	10	2
Bridges	106	11	7
Echocardiogram	132	12	2
Flag	194	28	8
Glass	214	10	7
Heart	270	13	2
Heart(Hungarian)	294	13	5
Heart(Long Beach VA)	200	13	5
Heart(Swiss)	123	13	5
Hepatitis	155	19	2
Heart(Cleveland)	303	13	5
Horse Colic	301	23	2
Image Segmentation	420	18	7
Ionosphere	351	34	2
Iris	150	4	3
Liver (Bupa)	345	6	2
New Thyroid	215	5	3
Pima Diabetes	768	8	2
Promoters	106	57	2
Sonar	208	60	2
Vehicle	846	18	4
Voting	435	15	2
Vowel	990	10	11
Wine	178	13	3
Zoo	90	16	7

Tabla 1. Características de las bases de datos empleadas en los experimentos

Para determinar qué tan buenos son los resultados de clasificación, es necesario estimar el porcentaje de aciertos después del proceso de clasificación. Una manera de evaluar los resultados de clasificación es mediante conjuntos de prueba y entrenamiento.

En los experimentos realizados en este reporte, los conjuntos de prueba y entrenamiento fueron construidos empleado validación cruzada (*k-fold cross validation*), específicamente *10 fold cross validation*, que consiste en dividir cada base de datos en 10 partes (de aproximadamente igual tamaño), de las cuales 9 se emplean como conjunto de entrenamiento (90% de los datos) y la parte restante (10% de los datos) se usa como conjunto de prueba. Cada una de las partes resultantes de la división de la base de datos se considera como conjunto de prueba, por lo que se realizan un total de 10 experimentos por cada base de datos.

Una vez que se obtuvieron los correspondientes conjuntos de prueba y entrenamiento, éstos se proporcionaron a cada uno de los métodos de edición, la edición se aplicó al conjunto de entrenamiento y la calidad del subconjunto obtenido mediante el proceso de edición, fue evaluada con el conjunto de prueba.

Para que los experimentos se llevaran a cabo en total igualdad, estos fueron realizados bajo las mismas condiciones, es decir, en cada método se emplearon los mismos conjuntos de prueba y entrenamiento, el mismo clasificador (*k-NN*) y se utilizó el mismo equipo de cómputo. Además, se utilizó la función heterogénea de distancia descrita en la ecuación 3, ya que algunas de las bases de datos empleadas contienen objetos con atributos numéricos y no numéricos.

Los resultados promedio obtenidos en los 10 experimentos realizados en cada base de datos se muestran en la tabla 2 aplicando los métodos de edición decremental: *ENN, RENN, DROP1-DROP5 y BSE*. También se incluye el resultado de clasificación usando *k-NN* considerando el 100% de los datos, es decir, la base de datos sin editar.

Para cada método, en la tabla 2 se muestra el porcentaje de clasificación correcta obtenido al evaluar la muestra editada que se obtuvo con cada método. También se muestra el porcentaje de retención (%) obtenido, es decir, el tamaño de la muestra editada con respecto al 100% de la original. En la parte inferior de la tabla 2 se incluye el promedio de los resultados obtenidos en las 25 bases de datos.

Base de datos	*k-NN*	%	ENN	%	RENN	%	DROP n										BSE	%
							1	%	*2*	%	*3*	%	*4*	%	*5*	%		
Breast cancer (WI)	96.28	100	96.42	96.58	96.28	96.41	93.13	2.85	96.28	5.80	95.42	3.26	95.99	3.70	95.56	4.18	98.71	2.09
Bridges	66.09	100	59.46	58.23	58.36	65.93	39.64	10.17	61.18	17.30	56.36	17.60	57.36	21.28	62.82	22.22	85.63	15.50
Echocardiogram	90.00	100	92.86	93.10	92.86	93.10	87.50	12.61	93.04	15.15	90.18	13.80	90.18	13.80	92.86	15.61	97.32	6.45
Flag	61.34	100	63.32	57.07	62.76	62.83	60.76	19.02	60.76	27.15	60.89	15.12	63.97	22.51	59.26	22.16	80.97	21.81
Glass	71.90	100	68.64	71.96	65.37	66.93	66.84	23.21	65.89	31.05	66.28	24.35	67.77	29.39	62.16	25.91	89.67	13.18
Heart	82.22	100	82.22	84.28	82.22	82.88	75.56	12.22	82.96	20.82	81.11	14.49	80.37	16.38	78.52	17.12	97.40	11.39
Heart(Hungarian)	79.55	100	80.28	82.12	79.25	79.93	72.17	5.74	78.52	8.80	80.84	12.76	78.19	15.26	79.84	15.37	94.27	14.88
Heart(Long Beach VA)	73.00	100	73.50	76.17	74.00	74.00	67.50	14.78	69.50	20.00	72.00	8.11	70.50	12.89	69.50	13.94	84.50	4.33
Heart(Swiss)	93.72	100	93.72	93.50	93.72	93.50	93.72	2.89	92.88	4.61	93.72	1.81	93.72	1.81	93.72	1.81	93.72	3.61
Hepatitis	80.62	100	81.25	83.73	80.58	82.80	72.38	4.66	80.75	10.54	81.87	7.81	78.75	9.75	83.29	9.39	97.41	9.24
Heart(Cleveland)	82.49	100	82.17	83.97	82.17	83.02	77.20	12.83	79.20	21.31	78.89	11.44	79.53	13.53	79.87	14.59	97.35	15.04
Horse Colic	66.08	100	67.09	66.70	67.09	65.26	61.76	6.72	66.75	16.43	66.41	3.25	65.44	7.49	65.41	9.60	92.02	13.91
Image Segmentation	92.86	100	92.14	92.46	91.90	91.61	86.19	11.01	92.62	13.52	90.24	10.93	91.67	12.28	91.43	11.64	97.61	8.22
Ionosphere	84.60	100	83.17	84.05	83.17	82.43	87.73	7.72	89.16	11.90	86.87	7.15	86.31	10.60	86.87	9.50	92.86	3.60
Iris	94.67	100	94.67	95.04	94.00	94.96	89.33	9.56	94.67	16.96	95.33	15.33	94.67	15.26	94.00	12.44	99.33	6.14
Liver (Bupa)	65.22	100	61.16	66.44	61.76	60.45	57.98	31.53	64.08	39.26	67.82	26.83	66.41	33.11	63.46	30.59	96.52	12.69
New Thytoid	95.39	100	91.69	95.09	91.69	94.57	85.56	8.32	90.78	14.88	93.98	9.77	93.51	10.39	94.46	8.84	97.70	3.61
Pima Diabetes	72.79	100	74.61	76.39	75.00	73.94	68.75	18.88	72.91	28.31	72.91	16.44	71.23	21.70	72.26	21.28	94.27	9.33
Promoters	93.36	100	92.45	96.12	92.45	95.81	89.64	6.50	87.36	17.09	90.27	16.46	90.45	16.98	85.91	10.38	100	16.25
Sonar	83.71	100	78.90	83.55	78.43	82.16	69.79	24.73	81.31	32.53	81.29	27.03	83.64	31.62	79.86	29.01	98.57	10.67
Vehicle	71.99	100	70.22	74.99	69.98	71.79	67.15	22.98	67.03	31.13	69.97	23.50	69.63	28.42	69.28	26.62	95.26	16.24
Voting	95.63	100	90.80	92.16	90.80	90.83	94.94	3.27	95.18	7.23	95.86	4.65	96.09	5.06	95.86	7.41	97.46	3.49
Vowel	97.47	100	92.73	95.94	91.82	95.38	82.26	35.14	92.22	41.17	88.99	39.71	91.31	40.93	91.01	35.82	99.49	12.33
Wine	95.00	100	93.89	95.69	93.89	95.69	94.97	10.36	95.52	14.00	94.41	15.04	94.41	15.04	93.86	10.55	100	4.62
Zoo	93.33	100	87.78	93.33	87.78	93.21	87.78	16.79	86.67	20.37	90.00	20.37	91.11	21.36	95.56	18.77	97.77	10.86
Promedio	83.17	100	81.80	83.54	81.49	82.77	77.20	13.37	81.48	19.49	81.67	14.68	81.68	17.22	81.46	16.19	95.03	9.97

Tabla 2. Porcentajes de clasificaciones correctas y retención obtenidos para: *k*-NN (100% de los datos), *ENN, RENN*, *DROP1 –DROP5* y *BSE.*

En base a los resultados obtenidos en la fase experimental, a continuación se presenta un análisis comparativo de acuerdo a los porcentajes de clasificación y reducción, para, de esta manera, determinar en qué sentido los métodos resuelven el problema de edición, es decir, si reducen el tamaño de la muestra sin mejorar la clasificación o resuelven el problema en ambos sentidos.

En lo que se refiere a clasificación, *ENN* resulta ser ligeramente mejor que *RENN* y de manera contraria ocurre en los resultados de retención, pues con *RENN*, en la mayoría de los casos, el porcentaje de retención es menor al obtenido con *ENN*.

En cuanto a los métodos *DROP,* con *DROP3* y *DROP4* se obtienen los mejores resultados de clasificación, y además mejoran un poco a los obtenidos con *ENN y RENN*. El menor porcentaje de objetos retenidos de entre los métodos *DROP* se obtiene con *DROP1*. Se puede notar que con los métodos *DROP, ENN, RENN* se soluciona el problema de edición sólo en términos de reducir la muestra sin considerar la calidad de clasificación, ya que los resultados de clasificación no superan y tampoco llegan a igualar a los obtenidos con la muestra original.

Con respecto a los resultados obtenidos con *BSE,* en lo que se refiere a clasificación, en la mayoría de los casos, son mejores a los obtenidos con *DROP3* y en cuanto a porcentaje de retención, éste es menor que el obtenido con *DROP1*, de tal manera que *BSE* además de reducir el tamaño de la muestra incrementa los resultados de clasificación e incluso mejora los obtenidos considerando el 100% de los datos.

5. Conclusiones

En un proceso de clasificación supervisada es importante la calidad que presenta el conjunto de entrenamiento, ya que en base a éste se realizarán las futuras clasificaciones, pero en la práctica los conjuntos de entrenamiento suelen incluir objetos que no aportan información de utilidad para los clasificadores, debido a esta razón surge la necesidad de editar las muestras de tal manera que se proporcione información al clasificador con la cual se mejore sus resultados.

El método propuesto en este reporte ofrece una solución alternativa al problema de edición, ya que aborda el problema en ambas direcciones: mejorar los resultados de la clasificación y reducir de manera considerable el tamaño de la muestra sin degradar los resultados de clasificación. Esta reducción es muy importante para clasificadores basados en instancias, ya que, al proporcionarles una muestra pequeña, reducirán el tiempo empleado en el proceso de clasificación. También se reducen los recursos necesarios para almacenamiento.

A pesar de que el método *BSE* es una técnica no exhaustiva, éste presenta un alto costo computacional, debido a que en cada paso se verifica la precisión de la clasificación al eliminar cada uno de los objetos en el conjunto parcial.

Debido a que esta primera versión del método *BSE* presenta el inconveniente referente al alto costo computacional, es importante, en un futuro, proponer alternativas para reducir el orden de complejidad del método.

Referencias

[1] Wilson, D.R. and Martínez T.R. (2000). Reduction Techniques for Instance-Based Learning Algorithms. *Machine Learning* .38, pp. 257-286.

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

[3] Hart, P. E. (1968). The Condensed Nearest Neighbor Rule. *IEEE Transactions on Information Theory*, *14*, pp. 515-516.

[4] Ritter, G. L., Woodruff, H.B., Lowry S.R. and Isenhour T. L. (1975). An Algorithm for a Selective Nearest Neighbor Decision Rule. *IEEE Transactions on Information Theory*, *21-6*, November 1975, pp. 665-669.

[5] Gates, G. W. The Reduced Nearest Neighbor Rule (1972). *IEEE Transactions on Information Theory,* IT-18-3, pp. 431-433.

[6] Wilson, D.L. (1972). Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, *2-3*, pp. 408-421.

[7] Tomek, I. (1976). An Experiment with the Edited Nearest-Neighbor Rule. *IEEE Transactions on Systems, Man, and Cybernetics*, *6-6*, pp. 448-452.

[8] Aha, D.W., Kibler, D., Albert, M.K. (1991). Instance-Based Learning Algorithms. *Machine Learning*, *6*, pp. 37-66.

[9] Aha, D. W. (1992). Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies*, *36*, pp. 267-287.

[10] Riquelme, J.C., Aguilar-Ruíz, J.S., Toro, M. (2003). Finding representatives patterns with ordered projections. *Pattern recognition*. 36, pp. 1009-1018.

[11] Sánchez, J.S., Barandela, R., Marqués, A.I., Alejo, R., Badenas, J. (2003). Analysis of new techniques to obtain quality training sets. *Pattern Recognition Letters,* 24, pp. 1015-1022.

[12] Yuan J. and Zhi-Hua Z. (2004). Editing Training Data for *k*-NN Classifiers with Neural Network Ensemble. *Advances in Neural Networks – ISNN 2004: International Symposium on Neural Networks, Dalian, China, August 2004. Proceedings, Part I*, pp. 356-361.

[13] Barandela, R. and Gasca, E. (2000). Decontamination of training data for supervised pattern recognition methods. In: *Advances in Pattern Recognition Lecture Notes in Computer Science 1876.* S*pringer-Verlag.* pp. 621-630.

[14] Koplowitz, J., Brown, T. A. (1981). On the relation of performance to editing in nearest neighbour rules. *Pattern Recognition, 13,* pp. 251-255.

[15] Chang, C.L. (1974). Finding prototypes for nearest neighbour classifiers. *IEEE Transactions on Computers, 23-11*, November 1974, pp. 1179-1184.

[16] Devijver, P. A. & Kittler, J. (1982). *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, NJ: Prentice/Hall.

[17] Covert, T.M. and Hart, P.E. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Computers, 13*, pp. 21-27.

[18] Stanfill, C., and Waltz D. (1986). Toward memory-base reasoning*, Communications of the ACM, 29*, pp. 213-1228.

[19] Ralambondrainy, H. (1987). A clustering method for nominal data and mixture of numerical and nominal data, *Proc. Fisrt International Conference. Federation of Classification Societies, Aachen.*

[20] Yang, M.S., Hwang, P., Y. and Chen, D., H. (2004). Fuzzy clustering algorithms for mixed feature variables. *Fuzzy Sets and Systems. 141-2*, pp. 301-317.

[21] Blake, C., Keogh, E., Merz, C.J. (1998). UCI repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Department of Information and Computer Science, University of California, Irvine, CA.