Detección de ruido y aprendizaje basado en información actual

Damaris Pascual González[1], Fernando Daniel Vázquez Mesa[1] y Jorge Luis Toro Pozo[2]

[1] Facultad de Ciencias Económicas y Empresariales, Universidad de Oriente, Santiago de Cuba, Cuba

[2] Facultad de Matemática y Computación, Universidad de Oriente, Santiago de Cuba, Cuba

{dpascual, fvazquez}@eco.uo.edu.cu, jorgetp@ult.edu.cu

Resumen. Los métodos de limpieza de ruido tienen una gran significación en tareas de clasificación y en situaciones en las que es necesario realizar un aprendizaje semi-supervisado, debido a la importancia que tiene contar con muestras bien etiquetadas (prototipos) para clasificar nuevos patrones. En este trabajo, presentamos un nuevo algoritmo de detección de ruido en flujos de datos, que tiene en cuenta los cambios de los conceptos en el tiempo (*concept drift*), el cual está basado en criterios de vecindad, y su aplicación en la construcción automática de conjuntos de entrenamiento. En los experimentos realizados se utilizaron bases de datos sintéticas y reales, las últimas fueron tomadas del repositorio UCI, los resultados obtenidos avalan nuestra estrategia de detección de ruido en flujos de datos y en procesos de clasificación.

Palabras clave. Limpieza de ruido, flujo de datos, aprendizaje semisupervisado; *concept drift*.

Noise Detection and Learning Based on Current Information

Abstract. Methods for noise cleaning have great significance in classification tasks and in situations when it is necessary to carry out a semi-supervised learning due to importance of having well-labeled samples (prototypes) for classification of the new patterns. In this work, we present a new algorithm for detecting noise in data streams that takes into account changes in concepts over time (concept drift). The algorithm is based on the neighborhood criteria and its application uses the construction of a training set. In our experiments we used both synthetic and real databases, the latter were taken from UCI repository. The results support our proposal of noise detection in data streams and classification processes.

Keywords. Cleansing noise, data streams, semi-supervised learning, concept drift.

1 Introducción

Son incontables las esferas de la vida real en las que es necesario realizar un proceso de clasificación. Por ejemplo, en el procesamiento digital de imágenes, el reconocimiento del habla, el diagnóstico médico, la clasificación de nuevas especies en biología, el tratamiento automático de bases de datos, la video-vigilancia inteligente, el reconocimiento de huellas, en balística, procesamiento de textos, etc.

Para realizar el proceso de clasificación se necesita un conjunto de muestras etiquetadas (prototipos) lo suficientemente representativas, para que sean capaces de emitir un juicio correcto acerca de la clase a la cual pertenece un nuevo objeto. Este conjunto de muestras etiquetadas se conoce en la literatura como conjunto de entrenamiento (*Training Set*, TS), el cual, en muchos problemas, debe ser preparado por un experto humano por lo que constituye siempre un proceso complicado y costoso.

El proceso de clasificación se puede realizar de tres maneras: supervisada, no supervisada y semi-supervisada [3]. Los algoritmos de clasificación supervisada operan usualmente sobre la información suministrada por un conjunto de muestras (TS), patrones, ejemplos o prototipos de entrenamiento que son asumidos como representantes de las clases, y los mismos poseen una etiqueta de clase correcta. Por otro lado, las técnicas de clasificación no supervisada o de agrupamiento (*clustering*) [9], se emplean para construir el conjunto de entrenamiento cuando no se dispone del mismo, es decir, cuando no existe conocimiento acerca de las etiquetas de los patrones, incluso puede suceder

que no se conozca el total de clases presentes en los datos. Los algoritmos de clasificación semi-supervisada o de aprendizaje semi-supervisado [2, 6, 10, 12, 15, 20] tienen como única información a priori pocas muestras de las clases presentes y cuentan con un conjunto numeroso de objetos no etiquetados que serán utilizados también en el proceso de clasificación.

Debido a que el costo de preparación de un conjunto de entrenamiento por parte de un especialista es alto, surgen estrategias para el tratamiento automático de los datos con diversos objetivos. Enmarcados en el proceso de clasificación, una de las dificultades que se pueden presentar con el procesamiento automático es la asignación de etiquetas incorrectas a los patrones analizados.

En procesos de aprendizaje semi-supervisado, por ejemplo, en los que se cuenta con un conjunto pequeño de objetos bien etiquetados y un conjunto amplio de objetos sin clasificar, a los cuales es necesario asignar alguna etiqueta, se pueden cometer errores que más tarde ocasionarán a su vez fallos en la clasificación de nuevos objetos, ya que aprender de datos mal clasificados perjudica la funcionalidad de los algoritmos de clasificación, lo que demuestra la necesidad de aplicar estrategias de detección y eliminación de objetos ruidosos en las bases de datos.

Muchos algoritmos de detección de ruido trabajan sobre conjuntos de datos estáticos (algoritmos de edición), éstos tienden a obtener un conjunto de prototipos eliminando *outliers*, y no tienen en cuenta los cambios que se pueden ocasionar con el transcurso del tiempo [5, 14, 16, 18, 19].

Actualmente, existe una creciente necesidad de colección, almacenaje y transmisión de información en forma de un flujo de datos. Algunos investigadores suponen que los datos coleccionados provienen de fuentes de calidad, sin preocuparse de la posible contaminación de estos, por tanto, no tienen en cuenta que las etiquetas podrían ser erróneas. Otra cuestión que se debe tener en cuenta es el problema de los cambios en la distribución de los datos que pueden ocurrir en el transcurso del tiempo, induciendo de alguna manera, un cambio radical en el concepto final, dando lugar a lo que se denomina cambio de concepto (*concept drift*) [1, 7].

Como consecuencia de esto, si el entorno donde el clasificador ha sido entrenado aparece el *concept drift*, el clasificador deberá ser entrenado otra vez, debido a los cambios en los conceptos, y será necesario recurrir al experto humano nuevamente para que reconstruya el conjunto de entrenamiento [4, 13], haciéndose el proceso extremadamente tedioso y muy costoso.

En [21] aparece un método para eliminar ruido en un flujo de datos, utilizando técnicas estadísticas como el margen de varianza máxima, y hace una comparación entre las técnicas de Filtrado Local (FL), Global (FG) y, Local y Global (FLyG). Tiene tres limitaciones fundamentales: 1) necesita introducir un parámetro α que indica el número de objetos que considera como ruidosos en la base de datos, esto en general, es un problema ya que es imposible conocer de antemano qué porcentaje de contaminación tienen los datos presentes en el flujo, 2) necesita evaluar la función que caracteriza el principio del margen de varianza máxima muchas veces, lo que hace el proceso costoso y 3) los mejores resultados de su algoritmo se obtienen con el Filtrado FLyG, por lo que hace falta desarrollar tanto el filtrado local como el global, lo cual indica un mayor número de procesos a ejecutar.

En el presente trabajo, mostramos una nueva estrategia para la detección de ruido en flujos de datos mediante criterios de vecindad para eliminar las limitaciones del método en [21], en nuestra propuesta se utiliza un *ensemble* de dos clasificadores y se comparan las estrategias de Filtrado Global y, Filtrado Local y Global. Además, se hace una propuesta de un esquema de aprendizaje semi-supervisado que utiliza en la etapa de filtrado de las muestras etiquetadas por el sistema, el método de detección de ruido en flujos de datos propuesto en este marco. Este artículo está estructurado de la siguiente forma: en la Sección 2 se muestra el método que se propone para la detección de ruido en flujos de datos teniendo en cuenta el *concept drift*, con una sub-sección en la que se explican las características y discusión de los experimentos realizados. En la Sección 3 aparece el esquema de aprendizaje semi-supervisado para construir

un conjunto de entrenamiento utilizando el algoritmo de filtrado presentado, así como una sub-sección con los resultados alcanzados. Finalmente, las conclusiones del trabajo en la Sección 4.

2. Estrategia para el filtrado de las muestras en flujos de datos

Los métodos de limpieza de ruido en general, usan clasificadores entrenados de una porción de los datos de entrenamiento, para justificar las muestras excluidas. Esto puede ser posible para datos estáticos, pero en flujos de datos, es necesario tener en cuenta que ellos están sujetos a cambios en las diferentes distribuciones, por lo que es necesario definir estrategias en las que esté presente esta problemática.

Se pueden efectuar tres variantes para filtrar el flujo de datos: 1) El Filtrado Local (FL) realiza la limpieza de los datos localmente dentro de cada bloque, sin necesitar ningún otro bloque de datos, es decir, trata cada bloque de datos como un conjunto estático y separa cada bloque S_i en dos conjuntos A_i y N_i (aceptados y no aceptados o ruidosos, respectivamente). La desventaja de este método es el número reducido de prototipos que tiene cada bloque, por lo que se pueden cometer errores, y detectar incorrectamente ejemplos que no son ruidosos como ruidosos. 2) El Filtrado Global (FG) utiliza clasificadores entrenados desde múltiples bloques para identificar el ruido. 3) En el Filtrado Local y Global (FLyG) se tienen en cuenta los objetos ruidosos según cada una de las otras dos estrategias.

En este trabajo, se modela el flujo de datos a través de los bloques que denotamos por F_i (i = 1, 2, … , H) de objetos etiquetados, provenientes de diversas fuentes. Para todos los objetos de cada bloque F_i se aplica una regla de clasificación, y se verifica si la etiqueta asignada al objeto coincide con la etiqueta que tiene originalmente, en caso que esto no ocurra, el objeto se considera ruidoso y es eliminado. Ésta es la idea central de nuestro método, lo que explica la importancia de utilizar una estrategia de clasificación adecuada.

Un problema de clasificación en general puede ser descrito en la siguiente forma:

Sean $(X,\theta)=\{(x_1,\theta_1),(x_2,\theta_2),...,(x_N,\theta_N)\}$ un conjunto de muestras etiquetadas (conjunto de entrenamiento), y x un nuevo objeto que se quiere asignar a alguna de las M clases existentes (C_1, C_2,…, C_M). Si $p(c_j / x)$ son las probabilidades a posteriori de cada una de las clases, podemos asignar a x la etiqueta c_i (i = 1, 2,…, M) si:

$$p(c_i / x) = \max_{1\le j\le M} p(c_j / x) \tag{1}$$

Una de las técnicas más empleadas para manejar los cambios de conceptos son los *ensembles* de clasificadores, mediante los cuales las salidas de varios clasificadores se combinan para tomar una decisión final. En nuestra estrategia de clasificación, empleamos también un *ensemble* de clasificadores, ya que se utiliza un producto de dos funciones p_1 y p_2 que representan estrategias de clasificación diferentes y luego se normaliza como se expresa en la Fórmula 2.

$$p(c_j / x) = \frac{p_1(c_j / x) * p_2(c_j / x)}{\sum_{r=1}^{M} p_1(c_r / x) * p_2(c_r / x)} \tag{2}$$

Dado un conjunto de entrenamiento E y x un objeto a clasificar, denotamos por U_x al conjunto que contiene los k objetos de E más cercanos a x (k vecinos más cercanos de x), además de los elementos de E que tienen a x entre sus k vecinos más cercanos (k = 1, 3). Entonces,

$$p_1(c_j / x) = \frac{\left|\left\{u \in U_x / \theta_u = c_j\right\}\right|}{\left|U_x\right|} \tag{3}$$

donde θ_u representa la etiqueta de clase del objeto u y |X| representa el cardinal del conjunto indicado.Por otro lado, para definir la función p_2 tuvimos en cuenta la cercanía del objeto x a cada una de las clases presentes, es decir,

$$p_2(c_j / x) = \frac{1}{\varepsilon + d(x, C_j)} \quad (4)$$

donde ε es un número positivo pequeño para evitar que se anule el denominador, C_i son las clases en E y $d(x, C_j) = \min_{y \in C_j} d(x, y)$.

La idea intuitiva de la Fórmula (2), es que se tiene en cuenta una vecindad del punto *x*, es decir, los objetos del conjunto de entrenamiento que están alrededor de *x*, bajo la suposición de que los puntos que rodean a *x*, y están muy cerca del mismo, deben tener la misma etiqueta que *x*. También, es bueno mencionar, que estamos utilizando la distancia de *x* a cada una de las clases existentes, de este modo, si *x* está bien etiquetado, la distancia a su propia clase debe ser la mínima y por tanto, p_2 debe ser máxima.

Es precisamente en el filtrado global, que se puede tener en cuenta el *concept drift*, que significa que se desechen todos los bloques anteriores a uno dado, anterior a su vez a F_i. La idea de esta propuesta se basa en el hecho que si hay distribuciones de los datos muy antiguas, es aconsejable no tenerlas en cuenta, porque podría provocar criterios falsos acerca de la situación actual. Así, en un filtrado global se utiliza un parámetro β para determinar cuántos bloques anteriores a F_i van a formar parte del conjunto de entrenamiento E:

$$E = \bigcup_{j=i-\beta}^{i-1} A_j \quad (5)$$

El algoritmo de detección de ruido en flujos de datos que se propone en este trabajo se expresa a continuación.

Es necesario hacer la observación, que el conjunto de entrenamiento se constituye con los elementos de los β bloques anteriores a F_i que ya han sido aceptados como no ruidosos, ya que como se sabe los bloques que llegan tienen una cantidad dada de objetos ruidosos (Fórmula 5).

Algoritmo	
Entrada:	Flujo de datos de muestras etiquetadas: Bloques F_i (i = 1, …, H) Número de vecinos: k
Salida:	Conjuntos de elementos aceptados (uno por cada bloque): A_i

1. Para *i* desde 1 hasta *H* hacer:
1.1. Cargar el bloque F_i
1.2. Inicializar los conjuntos de elementos ruidosos N_i y aceptados A_i como conjuntos vacíos: $N_i = A_i = \phi$
1.3. Para cada elemento *x* de F_i hacer:
1.3.1. Definir el conjunto U_x
1.3.2. Hallar la etiqueta c_j que maximice $p(c_j / x)$ empleando la Fórmula (2)
1.3.3. Si $c_j = \theta_x$ entonces *x* se considera aceptado y se inserta en A_i, esto es, $A_i = A_i \cup \{x\}$
1.3.4. En caso contrario *x* se considera ruidoso y se agrega en N_i, haciendo $N_i = N_i \cup \{x\}$

2.1. Resultados experimentales

En este epígrafe se muestran los resultados obtenidos de la experimentación realizada para probar el método propuesto, sobre bases de datos reales y sintéticas. Entre las bases de datos se tomaron G4 y G6, dos bases de datos sintéticas formadas por Gaussianas, la primera tiene 4 modos gaussianos con cierto grado de solapamiento, mientras que la segunda tiene 6 modos gaussianos, 4 de ellos muy solapados. El resto de las bases de datos fueron tomadas del *UCI Machine Learning Database Repository* [8]. En la Tabla 1 se exponen las principales características de estas colecciones de datos, mostrando el número de objetos, clases, atributos y el tipo de cada una de ellas.

Tabla 1. Descripción de las características de las bases de datos

Base de Datos	# Clases	# Objetos	# Atributos	Tipo
Cancer	2	683	9	Reales
German	2	1000	24	
Diabetes	2	768	8	
Page	5	5473	10	
Phoneme	2	5404	5	
Wave	3	5000	21	
Pendigit	10	10992	16	
Spam	2	4601	57	
Satimage	6	6435	36	
Letter	26	20000	16	
Shuttle	7	43500	9	
Texture	11	5500	40	
G4	4	4000	2	Sintéticas
G6	6	6000	2	

Para simular el flujo de datos, cada una de las bases de datos fue dividida en 10 bloques de manera aleatoria, manteniendo la distribución de probabilidades de las clases, y de cada bloque del flujo de datos se seleccionó de manera aleatoria un porcentaje (α = 10, 20, 30, 40, 50) de objetos a los que se les cambió su verdadera etiqueta de clase, y fueron etiquetados aleatoriamente en otra clase para simular la existencia de objetos ruidosos en la base de datos. Con cada valor de α se generaron cinco conjuntos diferentes de objetos mal etiquetados, haciendo este proceso lo más aleatorio posible. El resultado que aparece en las tablas es el promedio de las cinco ejecuciones realizadas del proceso indicado.

El conjunto U_x (*k* vecinos más cercanos de *x* y los objetos que tienen a *x* entre sus *k* vecinos más cercanos) se construye tomando los valores de *k* = 1, 3 para comparar los resultados.

Con el objetivo de conocer la efectividad del método de detección del ruido, se utilizó como medida de calidad la Precisión.

$$\text{Precisión} = \frac{\left|R \cap \overline{R}\right|}{\left|R\right|} \tag{6}$$

donde $\overline{R}$ es el conjunto de los objetos ruidosos detectados por el algoritmo y *R* es el conjunto de los elementos ruidosos reales.

En las Tablas 2 y 3 se muestran los porcentajes de precisión en la detección del ruido que se obtuvo para cada una de las bases de datos, es decir, el porcentaje de objetos verdaderamente ruidosos que el algoritmo detectó como ruidosos para los valores de α = 10, 20 y 30.

Nótese que en todos los casos el mayor porcentaje de aciertos se obtuvo con el Filtrado Global (en negrita), o sea, con el Filtrado FG se detecta el mayor porcentaje de objetos ruidosos, tanto cuando se considera un vecino como si se utilizan los tres vecinos más cercanos del objeto en análisis.

Este es un resultado importante ya que disminuye el costo de la detección de ruido, pues no sería necesario realizar simultáneamente para cada bloque del flujo de datos un filtrado local y un filtrado global.

Tabla 2. Precisión de la detección de ruido para las bases de datos: Cáncer, German, Diabetes, Page, Phoneme, Wave, Pendigit, Spam y Satimage

Base de Datos	Método	10%	20%	30%
Cáncer	FG-1	**0,900**	**0,857**	**0,700**
	FLyG-1	0,843	0,738	0,536
	FG-3	**0,933**	**0,849**	**0,679**
	FLyG-3	0,903	0,772	0,537
German	FG-1	**0,722**	**0,633**	**0,579**
	FLyG-1	0,554	0,490	0,408
	FG-3	**0,762**	**0,656**	**0,598**
	FLyG-3	0,638	0,521	0,425
Diabetes	FG-1	**0,789**	**0,707**	**0,579**
	FLyG-1	0,631	0,548	0,385
	FG-3	**0,809**	**0,728**	**0,609**
	FLyG-3	0,686	0,595	0,434
Page	FG-1	**0,881**	**0,750**	**0,623**
	FLyG-1	0,807	0,611	0,444
	FG-3	**0,917**	**0,803**	**0,671**
	FLyG-3	0,886	0,708	0,523
Phoneme	FG-1	**0,808**	**0,713**	**0,585**
	FLyG-1	0,691	0,550	0,393
	FG-3	**0,846**	**0,748**	**0,629**
	FLyG-3	0,760	0,623	0,458
Wave	FG-1	**0,825**	**0,726**	**0,634**
	FLyG-1	0,733	0,602	0,461
	FG-3	**0,850**	**0,771**	**0,667**
	FLyG-3	0,800	0,688	0,525
Pendigit	FG-1	**0,771**	**0,686**	**0,590**
	FLyG-1	0,794	0,600	0,429
	FG-3	**0,923**	**0,808**	**0,671**
	FLyG-3	0,888	0,721	0,515
Spam	FG-1	**0,771**	**0,686**	**0,590**
	FLyG-1	0,628	0,512	0,403
	FG-3	**0,797**	**0,713**	**0,615**
	FLyG-3	0,685	0,560	0,442
Satimage	FG-1	**0,864**	**0,761**	**0,643**
	FLyG-1	0,778	0,638	0,476
	FG-3	**0,901**	**0,819**	**0,692**
	FLyG-3	0,863	0,737	0,555

Obsérvese también, que cuando se tienen en cuenta los 3 vecinos de cada objeto (FG-3 o FLyG-3), los porcentajes son superiores, ya que se está utilizando una vecindad más amplia, lo cual garantiza una mayor precisión en la detección de objetos mal etiquetados.

Es de destacar, que sobre las bases de datos cáncer, Page, Pendigit, Satimage, Letter, Shuttle y G4, con un 10% de datos mal etiquetados, considerando tres vecinos, se hace una limpieza del 90% o más de los objetos ruidosos con el filtrado FG. Sobre la base de datos Shuttle, por ejemplo, se hace una limpieza de más del 98% del ruido.

Tabla 3. Precisión de la detección de ruido para las bases de datos: Letter, Shuttle, G4 y G6

Base de Datos	Método	10%	20%	30%
Letter	FG-1	**0,877**	**0,748**	**0,618**
	FLyG-1	0,800	0,628	0,456
	FG-3	**0,904**	**0,785**	**0,659**
	FLyG-3	0,866	0,691	0,521
Shuttle	FG-1	**0,948**	**0,877**	**0,752**
	FLyG-1	0,872	0,720	0,527
	FG-3	**0,986**	**0,961**	**0,862**
	FLyG-3	0,945	0,847	0,648
G4	FG-1	**0,857**	**0,732**	**0,605**
	FLyG-1	0,758	0,594	0,412
	FG-3	**0,906**	**0,785**	**0,648**
	FLyG-3	0,877	0,699	0,483
G6	FG-1	**0,863**	**0,749**	**0,624**
	FLyG-1	0,780	0,612	0,426
	FG-3	**0,896**	**0,795**	**0,661**
	FLyG-3	0,854	0,709	0,492

Para las bases de datos: Diabetes, Wave, Spam y G6, con un 10% de objetos ruidosos se detecta un 80% o más de los mismos. Sólo en el caso de la base de datos German, se obtuvieron porcentajes de detección de ruido inferiores.

Cuando tomamos $\alpha = 40$, se detectó alrededor del 50% de los objetos ruidosos, mientras que para $\alpha = 50$ fueron eliminados alrededor del 40% de los objetos mal etiquetados (Tablas 4 y 5),

Tabla 4. Precisión de la detección de ruido (α = 40, 50)

Base de Datos	Método	40%	50%
Cancer	FG-1	**0,549**	**0,363**
	FLyG-1	0,344	0,158
	FG-3	**0,541**	**0,357**
	FLyG-3	0,344	0,154
German	FG-1	**0,505**	**0,431**
	FLyG-1	0,320	0,252
	FG-3	**0,503**	**0,438**
	FLyG-3	0,333	0,268
Diabetes	FG-1	**0,528**	**0,428**
	FLyG-1	0,327	0,233
	FG-3	**0,537**	**0,428**
	FLyG-3	0,363	0,245
Page	FG-1	**0,492**	**0,368**
	FLyG-1	0,289	0,174
	FG-3	**0,514**	**0,374**
	FLyG-3	0,317	0,177
Phoneme	FG-1	**0,483**	**0,395**
	FLyG-1	0,278	0,196
	FG-3	**0,499**	**0,389**
	FLyG-3	0,310	0,197
Wave	FG-1	**0,532**	**0,423**
	FLyG-1	0,331	0,224
	FG-3	**0,547**	**0,422**
	FLyG-3	0,353	0,218
Pendigit	FG-1	**0,492**	**0,395**
	FLyG-1	0,272	0,153
	FG-3	**0,513**	**0,352**
	FLyG-3	0,316	0,154
Spam	FG-1	**0,492**	**0,395**
	FLyG-1	0,294	0,215
	FG-3	**0,497**	**0,396**
	FLyG-3	0,309	0,219
Satimage	FG-1	**0,525**	**0,394**
	FLyG-1	0,331	0,201
	FG-3	**0,550**	**0,389**
	FLyG-3	0,362	0,200

siempre que se aplica el filtrado global, lo que no ocurre con el filtrado FLyG con el cual los porcentajes de detección de ruido son mucho menores.

Es bueno aclarar, que si cerca de la mitad de los ejemplos de la base de datos son ruidosos, hay una gran confusión entre los objetos ruidosos y los objetos con una etiquetada de clase correcta, lo que hace extremadamente difícil determinar cuáles son los objetos realmente ruidosos, se pueden observar en las Tablas 4 y 5 los resultados obtenidos en este caso.

También, evaluamos nuestro algoritmo utilizando los conjuntos de objetos aceptados como conjuntos de entrenamiento, ya que estamos interesados en emplear nuestra estrategia en un esquema de aprendizaje semi-supervisado.

Para esto, se seleccionó un conjunto de patrones bien etiquetados como conjunto de prueba (*Test*), determinando el porcentaje de clasificación correcta que los objetos aceptados como no ruidosos proporcionan al etiquetar los ejemplos del conjunto de prueba, ya que uno de los objetivos de la limpieza de ruido es su empleo en esquemas de aprendizaje semi-supervisado, donde se tiene un número grande de objetos sin etiquetar, y es necesario clasificarlos y filtrarlos para eliminar los errores de clasificación.

En esta prueba, tomamos los conjuntos de objetos aceptados del filtrado con las estrategias: FG y FLyG como conjunto de entrenamiento para clasificar el conjunto de prueba y comparamos los resultados obtenidos.

Se realizaron además dos experimentos cuyos resultados aparecen en las Tablas 6 y 7, en las columnas nombradas SF (Sin Filtrado) y FP (Filtrado Perfecto), que significan: todos los bloques antes de ser filtrados, y, todos los bloques luego de haber eliminado el total de los objetos ruidosos, respectivamente.

En las Tablas 6 y 7 aparecen los porcentajes de clasificación correcta promedio con la aplicación del filtrado propuesto, utilizando los tres vecinos más cercanos, ya que esa fue la estrategia de mejores resultados en la detección del ruido.

La columna BD significa base de datos, el símbolo α representa el porcentaje de ruido presente, y las restantes cuatro columnas, muestran cada uno de los conjuntos de entrenamiento formados por el resultado de aplicar las estrategias explicadas anteriormente (SF, FG, FLyG y FP).

Tabla 5. Precisión de la detección de ruido (α = 40, 50)

Base de Datos	Método	40%	50%
Letter	FG-1	**0,482**	**0,364**
	FLyG-1	0,311	0,206
	FG-3	**0,495**	**0,367**
	FLyG-3	0,333	0,210
Shuttle	FG-1	**0,587**	**0,378**
	FLyG-1	0,330	0,168
	FG-3	**0,673**	**0,386**
	FLyG-3	0,400	0,167
G4	FG-1	**0,468**	**0,347**
	FLyG-1	0,252	0,141
	FG-3	**0,501**	**0,357**
	FLyG-3	0,297	0,151
G6	FG-1	**0,493**	**0,370**
	FLyG-1	0,284	0,165
	FG-3	**0,513**	**0,373**
	FLyG-3	0,314	0,170

En negrita, marcamos los valores más significativos del porcentaje de clasificación correcta. Los resultados indican que en un esquema de aprendizaje, en el que se etiquetan objetos desconocidos, hasta un 20% de error en el etiquetado puede ser subsanado eliminando un porcentaje alto de los objetos ruidosos, y así, los conjuntos de entrenamiento tendrían mayor calidad.

Puede verse además en las Tablas 6 y 7, que cuando hay un 10% de objetos ruidosos, sobre las bases de datos: Cancer, German, Diabetes, G4, G6, Page, Wave, Pendigit, Spam, Satimage, Shuttle y Texture, se obtiene un porcentaje de clasificación correcta superior o similar al que se obtiene cuando se realiza un filtrado perfecto.

Esto significa, que se puede confiar en la estrategia de detección de ruido para construir conjuntos de entrenamiento. Sólo con las bases

de datos Phoneme y Letter quedaron los porcentajes por debajo de los del filtrado perfecto, pero sin mucha diferencia en el caso del filtrado FG.

Tabla 6. Porcentaje de clasificación correcta de los conjuntos de entrenamiento para k = 3

BD	α	SF	FG	FLyG	FP
Cancer	10	0,871	**0,964**	**0,966**	0,956
	20	0,761	0,924	**0,940**	0,939
	30	0,680	0,867	0,918	**0,944**
	40	0,571	0,710	0,765	**0,951**
	50	0,503	0,528	0,524	**0,946**
German	10	0,597	**0,662**	**0,672**	0,637
	20	0,533	**0,636**	**0,637**	0,627
	30	0,500	0,595	0,603	**0,638**
	40	0,463	0,536	0,532	**0,638**
	50	0,418	0,432	0,428	**0,616**
Diabetes	10	0,616	**0,701**	**0,694**	0,665
	20	0,564	**0,651**	**0,660**	0,649
	30	0,520	0,609	0,620	**0,679**
	40	0,461	0,540	0,544	**0,668**
	50	0,408	0,421	0,433	**0,644**
G4	10	0,887	0,977	**0,987**	**0,987**
	20	0,791	0,931	0,965	**0,986**
	30	0,694	0,838	0,896	**0,987**
	40	0,600	0,693	0,746	**0,984**
	50	0,494	0,503	0,498	**0,989**
G6	10	0,829	**0,930**	**0,938**	0,919
	20	0,742	0,890	**0,924**	0,918
	30	0,650	0,805	0,860	**0,919**
	40	0,560	0,657	0,706	**0,920**
	50	0,474	0,480	0,481	**0,928**
Page	10	0,853	**0,935**	**0,935**	**0,938**
	20	0,754	0,895	0,915	**0,936**
	30	0,667	0,806	0,855	**0,935**
	40	0,565	0,664	0,700	**0,929**
	50	0,480	0,484	0,479	**0,935**
Phoneme	10	0,744	0,808	0,801	**0,813**
	20	0,666	0,774	0,781	**0,802**
	30	0,604	0,712	0,734	**0,804**
	40	0,525	0,595	0,620	**0,799**
	50	0,454	0,467	0,470	**0,792**

Tabla 7. Porcentaje de clasificación correcta de los conjuntos de entrenamiento para k = 3

BD	α	SF	FG	FLyG	FP
Wave	10	0,705	**0,798**	**0,804**	0,773
	20	0,629	0,764	**0,786**	0,762
	30	0,557	0,699	0,746	**0,759**
	40	0,493	0,592	0,637	**0,759**
	50	0,419	0,453	0,459	**0,763**
Pendigit	10	0,876	0,966	0,966	**0,976**
	20	0,783	0,928	0,948	**0,974**
	30	0,687	0,837	0,882	**0,971**
	40	0,588	0,684	0,728	**0,968**
	50	0,489	0,485	0,476	**0,965**
Spam	10	0,665	**0,725**	0,714	0,717
	20	0,606	0,699	0,697	**0,714**
	30	0,544	0,626	0,642	**0,701**
	40	0,488	0,544	0,559	**0,695**
	50	0,429	0,448	0,452	**0,687**
Satima-ge	10	0,777	**0,859**	**0,855**	**0,859**
	20	0,698	0,831	0,849	**0,861**
	30	0,618	0,757	0,795	**0,859**
	40	0,529	0,622	0,661	**0,856**
	50	0,443	0,432	0,423	**0,851**
Letter	10	0,762	0,817	0,776	**0,838**
	20	0,680	0,771	0,740	**0,825**
	30	0,602	0,686	0,667	**0,807**
	40	0,514	0,557	0,534	**0,794**
	50	0,424	0,398	0,360	**0,776**
Shuttle	10	0,894	**0,995**	**0,994**	**0,996**
	20	0,797	0,984	0,984	**0,994**
	30	0,696	0,937	0,945	**0,994**
	40	0,600	0,794	0,801	**0,994**
	50	0,497	0,502	0,504	**0,994**
Texture	10	0,855	**0,941**	0,938	**0,949**
	20	0,762	0,899	0,912	**0,945**
	30	0,671	0,813	0,847	**0,942**
	40	0,573	0,680	0,709	**0,930**
	50	0,480	0,483	0,465	**0,922**

Cuando existe un 20% de objetos ruidosos en las bases de datos, también los resultados alcanzados con el método propuesto para la detección de ruido son buenos. Por ejemplo, sobre las bases de datos: German, Diabetes, Wave, G6 y Shuttle, los porcentajes de clasificación correcta son superiores o similares a los obtenidos con un filtrado perfecto. Para el resto de las bases de datos, los porcentajes se pueden considerar adecuados por su significado, ya que al realizar un filtrado con nuestro método se logra eliminar objetos ruidosos y disminuir la talla del conjunto de entrenamiento, por lo que en la etapa de clasificación se utiliza una menor cantidad de objetos y los resultados son cercanos a los que se obtendrían si se pudieran detectar todos los objetos ruidosos existentes, lo cual en la práctica no es posible de realizar automáticamente.

Se pueden destacar los resultados que se han obtenido con las bases de datos: Cancer, G4, G6, Page, Pendigit, Shuttle y Texture, para las cuales, el porcentaje de clasificación correcta que proporcionan es igual o superior al 90% cuando hay un 20% o menos de error en las etiquetas de los objetos que forman los bloques. Esto garantiza que la estrategia de detección de ruido, es capaz de filtrar los bloques del flujo de datos de manera que los objetos aceptados como no ruidosos puedan ser empleados para clasificar objetos nuevos.

Obsérvese además, la diferencia de los porcentajes obtenidos después de la limpieza con relación a los obtenidos si no se aplica nuestra estrategia. Para la mayoría de las bases de datos, el porcentaje de clasificación correcta que se obtiene del conjunto de entrenamiento con ruido (sin aplicar el método de filtrado que aquí se propone) es por lo menos un 10% menor que cuando se utilizan los bloques filtrados.

Por ejemplo, sobre la base de datos cáncer, con un 10% de objetos ruidosos, sin aplicar nuestro método, el porcentaje de clasificación correcta que proporciona el conjunto de entrenamiento es de un 87%, sin embargo, después de haber detectado objetos ruidosos, el porcentaje de clasificación correcta aumenta hasta más de un 96%.

Este resultado tiene una gran significación ya que por un lado, muestra que la estrategia de detección de ruido es importante, por otro lado, el hecho de que se hayan eliminado objetos significa que se está etiquetando nuevos objetos con un menor número de prototipos, lo cual hace el proceso menos costoso computacionalmente, ya que no es necesario tener en memoria tantos datos, y en tercer lugar, demuestra que la estrategia de tener en cuenta el cambio de conceptos, proporciona la construcción de conjuntos de entrenamiento adecuados sin necesidad de utilizar todos los objetos del flujo de datos, sino solamente los de los últimos bloques.

El hecho de obtener buenos resultados cuando se tiene en cuenta el *concept drift*, además de la utilidad en sí que tiene este problema en la actualidad, es importante ya que se puede ir eliminando información no relevante en el contexto actual. Desde el punto de vista computacional es conveniente, ya que para realizar una clasificación, no es necesario utilizar todos los objetos que ya han sido procesados, sino los de la última generación.

También se puede mencionar el hecho de que cuando hay un porcentaje de error de 40% o 50% se detecta menor cantidad de objetos ruidosos, causado por la gran confusión que debe existir en este caso, pues habría casi el mismo número de objetos bien etiquetados que mal etiquetados. Igualmente, esto influye en los porcentajes de clasificación correcta. De todas formas hay una reducción de los objetos mal etiquetados después de haber efectuado la estrategia.

3. Aprendizaje semi-supervisado

En aprendizaje semi-supervisado, se tiene un conjunto pequeño E de muestras correctamente etiquetadas y un conjunto grande de objetos sin clase que necesitan ser etiquetados para luego ser utilizados como conjunto de entrenamiento, con el objetivo de clasificar nuevas muestras. Se considera que los objetos sin etiqueta llegan formando una secuencia de bloques G_1, G_2,…, G_H, y con algún clasificador, se asignan etiquetas a los objetos, modelando de esta forma un flujo de datos F_1, F_2,…, F_H.

Entre los elementos etiquetados de cada bloque existen algunos ruidosos debido a errores en la clasificación, lo que puede ocasionar la

aparición del cambio de conceptos. Una manera de detectar el *concept drift* es mediante la aplicación del método de detección de ruido utilizando únicamente los dos resultados más recientes como se explicó en la sección anterior. Nuestra propuesta de esquema de aprendizaje semi-supervisado es la siguiente:

Algoritmo

Entrada: Conjunto de entrenamiento E
Flujo de datos de muestras etiquetadas: Bloques G_i (i = 1, …, H)
Número de vecinos: k

Salida: Conjuntos de elementos aceptados (uno por cada bloque G_i): A_i

1. Para i desde 1 hasta H hacer:
1.1. Clasificar los elementos del bloque G_i obteniendo el bloque de objetos etiquetados F_i
1.2. Se aplica la estrategia de filtrado global al conjunto F_i y se construye un nuevo conjunto A_i de objetos que se consideran no ruidosos
1.3. El conjunto A_i es el conjunto de entrenamiento construido en cada etapa.

Con esta nueva propuesta, estamos tomando como conjunto de entrenamiento actual el conjunto A_i obtenido, a diferencia de otros esquemas de aprendizaje semi-supervisado [17]. Las ventajas de esto es que el tamaño de cada conjunto A_i es pequeño por lo que no habrá mucha información en memoria, por otro lado, si se construyen los sucesivos conjuntos de entrenamiento según van llegando los bloques de objetos no etiquetados mediante la unión de los conocimientos nuevos con los anteriores, puede llegar el momento en que el tamaño del conjunto de entrenamiento sea grande, lo que llevaría a la necesidad de aplicar algún algoritmo de condensado [11].

Por tanto, en dependencia de la funcionalidad del clasificador empleado en el paso 1.1, y de la aplicación del algoritmo de detección de ruido en el paso 1.2, así será la calidad del conjunto de entrenamiento A_i obtenido en cada etapa.

Para desarrollar el paso 1.2 la primera vez, se selecciona un conjunto inicial A_0 de datos bien etiquetados que constituyen la experiencia existente acerca de la distribución de las clases, que sirve como conjunto de entrenamiento para etiquetar los objetos de F_1 y luego, decidir cuáles de ellos fueron mal etiquetados. La segunda vez, el conjunto de entrenamiento será la unión de A_0 y A_1 y desde entonces se utilizarán los dos conjuntos de aceptados anteriores al bloque que se evalúa.

3.1. Resultados experimentales

Para la realización de nuestros experimentos utilizamos algunas de las bases de datos mencionadas en la sección anterior. Primero seleccionamos el conjunto de entrenamiento E y el conjunto A_0 para la detección de ruido, conservando la distribución de clases de cada base de datos. Seleccionamos aleatoriamente el conjunto de prueba (*Test*), y los bloques G_i para simular la llegada de los datos no etiquetados.
Como clasificador supervisado para etiquetar a los objetos de G_i empleamos una estrategia basada en centroides. Por medio del algoritmo k-Means dividimos cada una de las clases presentes en el conjunto de entrenamiento en 5 grupos y tomamos el centroide de cada grupo, y para cada x perteneciente a G_i, se asigna la etiqueta de la clase c_r si el centroide más cercano a x es alguno de los de dicha clase.

Los conjuntos A_i obtenidos en cada etapa se evalúan como en la sección anterior desde dos puntos de vista: la calidad de la limpieza del ruido y según la información que él puede emitir al etiquetar el conjunto *Test*. En la estrategia de ruido se realiza un filtrado global, con el objetivo de tener en cuenta el *concept drift*, utilizándose los dos conjuntos anteriores al que se evalúa en un momento concreto, tal como se indica en la Fórmula 5.

Los resultados que se muestran en las Tablas 8, 9 y 10 representan el promedio de los porcentajes obtenidos en cada una de las etapas sobre los diferentes conjuntos de muestras. Agregamos en nuestro análisis de la calidad del

conjunto de entrenamiento que se construye, la limpieza perfecta del ruido (LP), es decir, cuando se elimina de F_i todos y solamente los objetos ruidosos, y, una estrategia de edición, o lo que es lo mismo, el filtrado o limpieza local de F_i para realizar las comparaciones.

El conjunto F_i denota el conjunto resultante luego de haber etiquetado a todos los objetos del bloque G_i con la estrategia de los centroides por clase, por tanto, en la columna segunda de la Tabla 8, aparece el porcentaje de objetos no ruidosos que origina la aplicación de esa regla de clasificación. En las columnas tercera y cuarta aparece el porcentaje de objetos bien etiquetados después de haber aplicado a F_i el método de detección y eliminación de ruido, globalmente y localmente respectivamente.

Obsérvese en la Tabla 8 que para la mayoría de las bases de datos el filtrado local mejora la calidad del conjunto F_i, pero con el filtrado global se detecta una mayor cantidad de objetos ruidosos. Es necesario tener en cuenta que los resultados obtenidos se consideran acorde con la situación presente, ya que, según nuestro enfoque, para considerar un objeto como no ruidoso, las etiquetas asignadas tanto por el clasificador basado en centroides, como por el ensemble de clasificadores deben ser iguales, lo que no sucede en la sección anterior en la que los objetos del flujo de datos ya están etiquetados por un experto.

Las consecuencias del hecho de que los elementos de G_i no están etiquetados es que hay una doble incertidumbre en cuanto a la calidad de la clasificación, por tanto, el objeto será rechazado si no hay concordancia en las etiquetas asignadas, pero podría suceder que ambos clasificadores cometan el mismo error. De todas maneras, según se observa en la Tabla 8, la limpieza global teniendo en cuenta el *concept drift* es la que obtiene mejores resultados.

Las bases de datos Cancer, Pendigit, Shuttle, Texture y G6, tienen menos del 10% de objetos ruidosos, después del etiquetado del paso 1.1 del algoritmo, mientras que Satimage tiene casi un 20% de objetos mal etiquetados. Para la base de datos sintética G6 se observa el menor porcentaje de objetos no ruidosos en relación con F_i, debido quizás al solapamiento tan grande que hay entre 4 de sus clases.

Obsérvese que en casi todos los casos, luego del filtrado global quedan menos de un 5% de objetos ruidosos, y en el caso de Satimage que tiene casi un 20% de elementos mal etiquetados después de la clasificación del paso 1.1, logra con la estrategia global eliminar un 7% de estos objetos del bloque F_i. En el caso de la base de datos Texture, también hay un porcentaje de eliminación considerable al lograr detectar casi un 5% de los elementos mal clasificados.

Tabla 8. Porcentaje de objetos no ruidosos luego de haber aplicado la estrategia

Base de Datos	F_i	A_i-Global	A_i-Local
Cancer	93,179	95,780	95,314
Pendigit	93,035	95,938	95,209
Shuttle	92,268	95,341	92,814
Texture	90,44	95,068	94,149
Satimage	80,95	87,761	85,051
G6	93,822	94,693	94,850

En las Tablas 9 y 10 se muestra la evaluación de la calidad de cada conjunto A_i como conjunto de entrenamiento al etiquetar el conjunto *Test*.

Tabla 9. Porcentaje de clasificación correcta del conjunto Test

Base de Datos	F_i	A_i-Global	A_i-Local
Cancer	93,835	95,529	94,576
Pendigit	93,23	94,626	93,7
Shuttle	92,380	94,611	92,514
Texture	89,926	92,256	90,79
Satimage	80,47	83,87	81,363
G6	93,000	93,218	93,166

En la Tabla 9 aparecen los resultados al tomar como conjunto de entrenamiento el conjunto resultante del etiquetado de los bloques G_i, luego del filtrado global y después del filtrado local, en las columnas segunda, tercera y cuarta, respectivamente.

Así, el resultado A_i, tiene la información actual para etiquetar muestras que llegan en un nuevo contexto. En el caso del filtrado local, se

observan porcentajes menores, ya que no utiliza la información existente en bloques anteriores que pueden enriquecer el conocimiento presente. Nótese en la Tabla 9, que no hay diferencias significativas entre el resultado del filtrado local y el de la clasificación en el paso 1.1, debido quizás al estrecho marco en el que se evalúan los objetos, recuérdese que son bloques de tamaño pequeño, por tanto, en un filtrado local los resultados se afectan.

En la Tabla 10, segunda columna, aparece G_i (0% error), la cual se refiere a los objetos del bloque G_i etiquetados correctamente, es decir, los bloques con las etiquetas originales de cada uno de los elementos que lo componen.

Claramente, los porcentajes de clasificación correcta que inducen los objetos bien etiquetados son más altos con relación a los mostrados en la Tabla 9, ya que, por un lado, el conjunto de entrenamiento es mayor, y por otro todos los objetos están bien etiquetados. De todas formas, los resultados obtenidos con la aplicación de la estrategia de ruido aquí propuesta, son satisfactorios, ya que, como se sabe, no se puede lograr una limpieza total del ruido, además, para detectar los objetos ruidosos con el filtrado global, solo se consideraron los dos últimos resultados anteriores al bloque actual, por lo que se desechó información, ya que la consideramos no relevante.

Tabla 10. Porcentaje de clasificación correcta del conjunto Test

Base de Datos	G_i (0% error)	LP
Cancer	95,793	96,031
Pendigit	97,813	96,095
Shuttle	99,571	97,085
Texture	94,716	94,02
Satimage	85,876	85,026
G6	91,572	93,458

En la última columna de la Tabla 10, aparecen los resultados de la clasificación del conjunto *Test* después de un filtrado perfecto de F_i. Se puede observar que los porcentajes son superiores a los mostrados en la Tabla 9, debido a que no existen elementos ruidosos y además, se han desechado exclusivamente éstos. Esto conlleva a la presencia de mayor información para realizar la tarea de clasificación de nuevas muestras como es de esperar.

En el caso de Cancer y G6 los porcentajes son superiores a los mostrados en la segunda columna de la Tabla 10, a pesar de que hay menor cantidad de objetos debido a la eliminación de los objetos ruidosos.

4. Conclusiones

En este trabajo se ha mostrado un nuevo enfoque para tratar el problema de la presencia de ruido en flujos de datos, empleando criterios de vecindad. En la nueva estrategia se utiliza un *ensemble* de dos clasificadores, para combinar los resultados que cada uno aporta en la etapa de clasificación. Este método se enfoca en el problema de la presencia del *concept drift*.

El método propuesto detecta automáticamente todos los objetos que considera ruidosos, no se limita a un porcentaje α (este valor sólo se utiliza para simular la existencia de objetos ruidosos en el flujo de datos). Se emplea una estrategia muy simple (vecinos más cercanos).

En los experimentos realizados para evaluar la estrategia de detección de ruido en flujos de datos, se tuvo en cuenta para cada objeto la vecindad de su vecino más cercano o la de sus tres vecinos más cercanos.
Se realizaron los experimentos siguiendo los esquemas de los filtrados: FG y FLyG debido a que con el filtrado local (FL) no se tiene en cuenta el *concept drift*.

Como medida para establecer la calidad del proceso de limpieza de ruido se utilizó la precisión, analizándose el porcentaje de objetos ruidosos que el algoritmo detecta y, la calidad de los bloques luego del proceso de limpieza, para ser utilizados como conjuntos de entrenamiento en la clasificación de nuevas muestras.

De las dos estrategias de filtrado, los resultados en el procesamiento de los patrones demuestran que el filtrado FG es suficiente para detectar los objetos ruidosos. Esto es importante ya que así el proceso es menos costoso debido a que no hay que realizar el filtrado local.

El método más efectivo resultó ser el que se realiza tomando los tres vecinos más cercanos de cada objeto, lo cual demuestra que para detectar los objetos ruidosos es más conveniente verificar las etiquetas de otros objetos que rodean al que se está analizando, no sólo su vecino más cercano.

Otra cuestión a destacar es que para tener en cuenta el cambio de conceptos, sólo se emplearon dos bloques anteriores al analizado en cada etapa, esto contribuye con un ahorro computacional importante, además del hecho en sí que es tener en cuenta nada más los resultados más actuales para detectar nuevos objetos ruidosos o para clasificar objetos correctamente, desechando información fuera del contexto actual.

Los porcentajes alcanzados en cuanto al filtrado de objetos ruidosos, demuestran la validez del método aplicado, ya que se detecta un 80% o más de individuos mal etiquetados cuando hay hasta un 20% de error de clasificación. La importancia de este hecho está en la posibilidad de emplear el método en esquemas de aprendizaje semi-supervisado.

En cuanto a la calidad como conjuntos de entrenamiento de los bloques filtrados, los resultados en los casos de menos de un 30% de ruido son alentadores. Los mejores resultados se obtienen cuando hay un 10% de ruido, ya que los porcentajes son superiores a los que se obtienen con el filtrado perfecto y se demuestra que con un menor número de objetos se obtienen porcentajes de clasificación satisfactorios.

Como una aplicación de los resultados obtenidos en el esquema de detección de ruido en flujos de datos, se propuso un algoritmo de aprendizaje semi-supervisado para desechar los objetos ruidosos producto de la etapa de clasificación. En este nuevo enfoque, se obtuvieron conjuntos de entrenamiento pequeños de buena calidad.

Referencias

1. **Bose, R.P.J.C., van der Aalst, W.M.P., Žliobaitė, I., & Pechenizkiy, M. (2011).** Handling concept drift in process mining. *Advanced Information Systems Engineering. Lecture Notes in Computer Science,* 6741, 391–405.
2. **Chapelle, O., Schölkopf, B., & Zien, A. (2006).** *Semi-supervised learning*. Cambridge, Mass.: MIT Press
3. **Duda, R.O., Hart, P.E., & Stork, D.G. (2001).** *Pattern Classification* (2nd ed.). New York: Wiley.
4. **Elwell, R. & Polikar, R. (2011).** Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10),1517–1531.
5. **García, S., Derrac, J., Cano, J., & Herrera, F. (2012).** Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3), 417–435.
6. **Kalish, C.W., Rogers, T.T., Lang, J., & Zhu, X. (2011).** Can semi-supervised learning explain incorrect beliefs about categories? *Cognition*, 120(1),106–118.
7. **Klinkenberg, R. (2004).** Learning drifting concepts: examples selection vs. examples weighting. *Intelligence Data Analysis*, 8(3), 281–300.
8. UCI Repository of Machine Learning Databases, University of California Irvine.
9. **Pascual D., Pla, F., & Sánchez, J.S. (2010).** A Density-based Hierarchical Clustering Algorithm for Highly Overlapped Distributions with Noisy Points. *2010 Conference on Artificial Intelligence Research and Development: Proceedings of the 13th International Conference of the Catalan Association for Artificial Intelligence*, Tarragona, Spain, 183–192.
10. **Qiuhua, L., Xuejun, L., Hui, L., Stack, J.R., & Carin, L. (2009).** Semisupervised Multitask Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6), 1074–1086.
11. **Rico-Juan, J.R. & Iñesta, J.M. (2012).** New rank methods for reducing the size of the training set using the nearest neighbor rule. *Pattern Recognition Letters,* 33(5), 654–660.
12. **Rohban, M.H. & Rabiee, H.R. (2012).** Supervised neighborhood graph construction for semi-supervised classification. *Pattern Recognition*, 45(4), 1363–1372.

13. **Ross, G.J., Adams, N.M., Tasoulis, D.K., & Hand, D.J. (2012).** Exponentially weighted moving average charts for detecting concept drift. *Pattern Recognition Letters,* 33(2), 191–198.
14. **Segata, N., Blanzieri, E., Delany, S.J., & Cunningham, P. (2010).** Noise reduction for instance-based learning with a local maximal margin approach. *Journal of Intelligent Information Systems*, 35(2), 301–331.
15. **Settles, B. (2009).** *Active learning literature survey* (Technical Report 1648). Madison, USA: University of Wisconsin.
16. **Vázquez, F., Sánchez, J.S., & Pla, F. (2005).** A stochastic approach to Wilson's editing algorithm. *Pattern Recognition and Image Analysis. Lecture Notes on Computer Science*, 3523, 35–42.
17. **Vázquez, F.D., Sánchez, J.S., & Pla, F. (2008).** Learning and forgetting with local Information of new objects. *Progress in Pattern Recognition, Image Analysis and Applications. Lecture Notes on Computer Sciences*. 5197, 261–268.
18. **Wilson, D.L. (1972).** Asymptotic properties of nearest neighbor rules using edited data*. IEEE Transactions on Systems Man and Cibernetics,* SMC-2(3), 408–421.
19. **Wilson, D.R. & Martínez, T.R. (2000).** Reduction techniques for instance based learning algorithms. *Machine Learning,* 38(3), 257–286.
20. **Zhou, Y. & Goldman, S. (2004).** Democratic Co-learning. *16th IEEE International Conference on Tools with Artificial Intelligence*, Boca Raton, Florida, 594–602.
21. **Zhu, X., Zhang, P., Wu, X., He, D., Zhang, C., & Shi, Y. (2008).** Cleansing Noisy Data Streams. *Eighth IEEE International Conference on Data Mining,* Pisa, Italy, 1139–1144.

Damaris Pascual González recibió el título de Doctora en Marzo de 2010 en la Universidad Jaume I de España. Es profesora de Matemáticas de la Facultad de Ciencias Económicas y Empresariales de la Universidad de Oriente, de Santiago de Cuba, Cuba. Actualmente trabaja en la temática Reconocimiento Estadístico de Formas.

Fernando Daniel Vázquez Mesa recibió el título de Doctor en Marzo de 2008 en la Universidad Jaume I de España. Es profesor de Matemáticas de la Facultad de Ciencias Económicas y Empresariales de la Universidad de Oriente, de Santiago de Cuba, Cuba. Actualmente trabaja en la temática Reconocimiento Estadístico de Formas.

Jorge Luis Toro Pozo se graduó de Licenciado en Ciencia de la Computación en Junio de 2012 en la Universidad de Oriente de Santiago de Cuba. Participó como programador del Grupo de Desarrollo de Software de la Facultad de Ciencias Técnicas de la Universidad de Las Tunas. Actualmente es profesor de Sistemas Operativos para la carrera Ingeniería Informática de la Universidad de Las Tunas, Cuba.

Artículo recibido el 22/04/2013, aceptado el 20/07/2013.