WIRS. Un Algoritmo de Reducción de Instancias Basado en Ranking *

Carlos G. Vallejo, José A. Troyano, and F. Javier Ortega

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Av. Reina Mercedes s/n 41012, Sevilla
vallejo@lsi.us.es

Resumen En este artículo se presenta el algoritmo WIRS, una técnica de reducción de instancias que tiene como objetivo seleccionar las instancias más representativas de una base de datos de aprendizaje. Este tipo de técnicas se utilizan para conseguir bases de datos más pequeñas sobre las que se pueda aplicar el algoritmo de los vecinos más cercanos con menor coste computacional y sin excesiva pérdida de precisión. El algoritmo WIRS es una adaptación del algoritmo WITS en el que se ha sustituido el criterio de la tipicidad por el de ranking a la hora de calcular el orden de las instancias necesario para aplicar WITS. Para calcular el ranking utilizamos una solución similar a la empleada por PageRank, el algoritmo de cálculo de relevancia de páginas web del buscador Google. Los experimentos demuestran que el uso del ranking como criterio de ordenación obtiene resultados comparables a los obtenidos por la versión original de WITS, mejorando incluso estos resultados para algunas de las bases de datos utilizadas.

Key words: Reducción de Instancias, Vecino más Cercano, PageRank, InstanceRank, WITS.

1. Introducción

La técnica de los vecinos más cercanos consigue muy buenos resultados de clasificación mediante un algoritmo muy simple. En esta técnica no hay una etapa de aprendizaje propiamente dicha, ya que no se llega a construir ningún modelo. Cada vez que se clasifica una instancia, ésta es comparada con todas las instancias de la base de datos de aprendizaje y se decide la clase que le corresponde en función de las clases de las instancias más parecidas. Esta comparación con todos los elementos de la base de datos permite que el algoritmo de clasificación tenga en cuenta regiones en el espacio de búsqueda que serían omitidas por otros algoritmos que sí realizan una generalización a partir de los datos. Pero no todos son beneficios, el coste de disponer de una técnica simple y potente se encuentra en los requerimientos de almacenamiento (al no existir modelo hay que

* Este trabajo ha sido parcialmente financiado por el Ministerio de Educación y Ciencia (TIN 2004-07246-C03-03)

mantener todas las instancias) y de tiempo (hay que comparar cada instancia a clasificar con todas las de la base de datos) en la etapa de clasificación.

Para resolver este problema se han propuesto varias soluciones, que se pueden organizar en dos categorías: las orientadas a reducir el número de instancias de la base de datos sin perder calidad en la clasificación ([1], [2], [3]) y las que buscan acelerar la búsqueda de las instancias más parecidas con la ayuda de estructuras de datos e índices ([4], [5]). Nuestro trabajo se enmarca dentro de la primera de las categorías, conocida como reducción de instancias, y consiste en una adaptación del algoritmo WITS.

En contra de lo que pudiera parecer, cuando se aplica una técnica de reducción de instancias no siempre es en perjuicio de la precisión del clasificador. En ocasiones, se clasifica mejor con el conjunto de instancias reducido que con la base de datos original. En estos casos se dice que la técnica edita la base de datos, ya que puede eliminar instancias ruidosas.

El algoritmo WITS se apoya en el concepto de tipicidad, que mide lo representativa que es una instancia para su clase. Todas las instancias de la base de datos de entrenamiento se ordenan según su tipicidad y esta ordenación es usada por el algoritmo para determinar el orden en el que se van procesando las instancias. Nuestra propuesta consiste en sustituir el concepto de tipicidad, y la ordenación que se deriva a partir de él, por un ranking calculado con técnicas similares a las utilizadas en el cálculo de la relevancia de los nodos de un grafo. En concreto, hemos utilizado una adaptación del algoritmo PageRank, que aporta uno de los criterios en los que se apoya el buscador Google para medir la relevancia de una página web en internet.

El resto del artículo se organiza de la siguiente forma. En la sección segunda se presenta el algoritmo WITS. La sección tercera presenta las ideas básicas de PageRank y el algoritmo para calcular el índice de relevancia de las instancias. La sección cuarta contiene nuestra propuesta: el algoritmo de selección de instancias WIRS que integra las ideas del algoritmo WITS con una adaptación de PageRank (que hemos denominado InstanceRank) que permite crear un ranking de instancias; también se incluye en esta sección el diseño experimental y los resultados obtenidos. Por último, en la sección quinta se extraen las conclusiones y se presentan las líneas de trabajo futuro.

2. WITS: Selección de instancias basadas en la tipicidad

WITS (*Weighted Instance Typicality Search*) es una técnica de selección (y, por tanto, de reducción) de instancias propuesta por Morring y Martinez [6]. Se basa en el concepto de *tipicidad* de una instancia, propuesto por Zhang [7] como una medida de la representatividad de una instancia dentro de una clase: es el cociente entre la similitud media de la instancia con todas las de su misma clase (similitud intra-clase) y la similitud media con todas las de distinta clase (similitud extra-clase), donde la similitud de dos instancias x e y se define como $1 - dist(x, y)$. Característicamente, las instancias centrales de un cluster tendrán una mayor similitud intra-clase, y de ahí una mayor tipicidad, los puntos de

la frontera tendrán una tipicidad intermedia y los elementos que conformen ruido o que sean excepcionales tendrán una tipicidad más baja: la tipicidad es un concepto relativo. WITS es un algoritmo de orden $O(n^2)$. Produce un gran suavizado de los bordes de las clases, y puede conseguir cubrir de una forma bastante completa problemas en los que la frontera de decisión sea muy compleja.

En WITS se toma como entrada el conjunto de instancias de entrenamiento T y se devuelve el conjunto S de instancias en el que cada una lleva un peso asociado, del siguiente modo: se considera el conjunto de instancias del conjunto de entrenamiento T dividido en *cubetas*, cada una de ellas formadas por las instancias de una clase. A cada paso del algoritmo, y mientras queden instancias y haya errores, se considera una instancia *candidata* de la cubeta que contenga el mayor número de instancias mal clasificadas por las instancias seleccionadas hasta el momento, y dentro de ella, la instancia restante de mayor tipicidad. Se calcula un peso óptimo de *candidata* de manera que minimice el número de errores producido por $S \cup candidata$.

Si ese error mejora sustancialmente el error que había hasta el momento, se selecciona *candidata*, añadiéndola a S, asociándole el peso óptimo calculado anteriormente y recalculando el número de errores de cada cubeta. Una especial mención merece la manera de evaluar cuándo se mejora sustancialmente el error y cómo se calcula el peso óptimo: se considera que una instancia *candidata* mejora sustancialmente el error anterior si el error posterior, supuesto que se añadiera la instancia, es menor que un cierto parámetro de la aplicación G, de modo que las instancias ruidosas o las que producirían un sobreajuste perjudicial no se añaden. Este parámetro suele tomarse proporcional al tamaño del conjunto de entrenamiento. El peso óptimo se calcula tomándolo de una serie de posibles valores de la forma b^k; en [6] se considera $b = 1{,}1$ y $k = (0, 1, 2, \ldots, 20)$, de modo que los pesos candidatos son [1.0, 1.1, 1.21, 1.331, ...,6.116, 6.727].

En la evaluación del error se considera el vecino más cercano, $k = 1$. La distancia utilizada es HVDM (*Heterogeneous Value Difference Metric*), presentada en [8], que usa la distancia euclídea para los atributos continuos y *VDM* para los nominales, que es una métrica que determina la distancia entre los valores de los atributos según la proximidad que tengan las clases que los suelen acompañar.

En la fase de generalización se utiliza la distancia anterior, ponderando las distancias según los pesos asignados a las instancias. Se elige como clase de la instancia a generalizar la de la clase del vecino más cercano ($k = 1$), según esa distancia ponderada.

La estructura general del algoritmo WITS es la siguiente:

func WITS(T: conjunto de instancias) **dev** S: conjunto de instancias ponderadas
algoritmo
    para cada instancia *ins* en T:
        calcula la tipicidad de *ins*
    fin para
    ordena las instancias en orden descendente de tipicidad
    para cada instancia *ins* en T:

```
        asigna ins a la cubeta de su clase, manteniendo el orden
    fin para
    S := ∅
    numErroresAnterior := |T|
    para cada cubeta cubeta:
        numErroresCubeta := numero de elementos en cubeta
    fin para
    mientras haya instancias en las cubetas y numErroresAnterior > 0:
        sea sigCubeta la cubeta con mayor numErroresCubeta
        cand := siguiente instancia de sigCubeta
        elimina cand de sigCubeta
        pesoCand := peso que minimiza el número de errores
        numErroresPosterior := error obtenido por cand ∪ S evaluado sobre T
        si numErroresAnterior − numErroresPosterior ≥ G:
            asigna el peso pesoCand a cand
            S := cand ∪ S
            numeroErroresAnterior := numErroresPosterior
            para cada cubeta:
                recalcula numErroresCubeta
            fin para
        fin si
    fin mientras
fin algoritmo
```

3. PageRank

El PageRank, presentado por Brin y Page en [9], es la medida de la importancia de una página web en internet utilizada por el buscador Google. Si consideramos Internet como un grafo dirigido, en el que los nodos son las páginas web y la existencia de una arista entre dos nodos denota que hay un enlace del primero hacia el segundo, se puede extraer un índice de relevancia de cada nodo sin más que considerar la topología del grafo (aunque Google realmente tiene en cuenta muchos más factores). Este índice de relevancia es el PageRank, y se basa, como sustrato teórico, en la existencia de un "navegador aleatorio"que va visitando páginas web pulsando los enlaces que ve en ellos, o bien se va a una página totalmente distinta. Esto se traduce en lo siguiente: si llamamos $PR(V)$ al PageRank de la página V, y llamamos $In(V)$ al conjunto de las páginas que tienen enlaces hacia V (en la terminología de grafos, los vértices origen de las aristas que tienen como destino V), y $Out(V)$ el conjunto de los vértices hacia los que tiene enlace V (los vértices destino de las aristas que tienen como origen a V) es

$$PR(V) = (1-d) + d \sum_{W \in In(V)} \frac{PR(W)}{|Out(W)|}$$

donde d es un valor entre 0 y 1 ([9] recomienda 0.85) que modela la probabilidad de que el "navegador aleatorio"navegue por los enlaces de una página y $(1-d)$ la probabilidad de que vaya a una página cualquiera al azar.

El PageRank de cada página web puede calcularse mediante un sencillo algoritmo iterativo y representa una distribución de probabilidad sobre las páginas web.

El TextRank, debido a Mihalcea [10], es una adaptación del PageRank a un problema muy distinto al anterior: la extracción de palabras clave y resúmenes de textos en el ámbito del lenguaje natural. Una palabra o frase (en general, un vértice de un grafo) tiene un ranking mayor o menor dependiendo de la influencia que sobre ésta tengan las demás; a este ranking se le denomina TextRank. En este contexto, una palabra o frase puede influir más o menos sobre otra, dependiendo de su similitud; por esta razón, y a diferencia del PageRank, la influencia tiene una cierta ponderación: un vértice V_i influye en un vértice V_j con un peso w_{ij} (por tanto, el grafo es ponderado).

La expresión del PageRank anterior se convierte en el TextRank en

$$TR(V_i) = (1-d) + d \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_j)} w_{jk}} TR(V_j)$$

Por otra parte, la influencia de una palabra o frase sobre otra es recíproca, de ahí que el grafo que describe la situación sea no dirigido ($w_{ij} = w_{ji}$, $In(V) = Out(V)$), lo que simplifica mucho los cálculos.

La expresión del TextRank, al igual que la del PageRank, puede calcularse mediante un cálculo iterativo, que converge.

4. WIRS: Selección de instancias basada en ranking

En este trabajo presentamos el algoritmo WIRS (*Weighted Instance Ranking Search*). En esta propuesta hemos partido de la premisa de que la medida de relevancia que aporta el ranking puede ser útil para determinar qué instancias son más importantes dentro de la base de datos, de la misma forma en la que PageRank mide la importancia de una página web en internet o TextRank la de una palabra o frase dentro de un texto. Por otra parte, hemos utilizado la estructura general del algoritmo WITS en cuanto a la mecánica del orden de selección de las instancias para su inclusión o no en el clasificador: se consideran las cubetas en orden descendente de número de errores y dentro de cada una de ellas según su ranking. También hemos utilizado de WITS el criterio de seleccionar una instancia sólo si su inclusión en el conjunto de instancias mejora sustancialmente las prestaciones del clasificador.

4.1. InstanceRank

Hemos considerado las instancias de una base de datos como vértices de un grafo. Cada arista de este grafo será la similitud entre las instancias que forman los extremos de la arista, que está etiquetada con esa similitud. Este grafo, por su naturaleza, es completo y no dirigido. Hemos definido la similitud entre dos instancias como una función de la distancia entre esas dos instancias. Para la distancia entre dos instancias hemos utilizado la distancia euclídea para los atributos continuos. Para los discretos hemos considerado distancia 0 para los que tienen el mismo valor y 1 para los que tienen distinto valor. Finalmente, la distancia se ha normalizado, de manera que

$$\forall v_1, v_2 \in T \quad 0 \leq \mathrm{dist}(v_1, v_2) \leq 1$$

Posteriormente hemos calculado la similitud entre dos instancias como una medida que cumple que $\forall v_1, v_2, v_3 \in T$:

$$sim(v_1, v_1) = 1$$
$$sim(v_1, v_2) = sim(v_2, v_1)$$
$$sim(v_1, v_2) \geq sim(v_1, v_3) + sim(v_3, v_2)$$

Como funciones de similitud hemos experimentado con las siguientes:

$$sim(dist) = 1 - \frac{dist}{k}$$
$$sim(dist) = \frac{1}{1 + k\, dist}$$
$$sim(dist) = e^{-k \frac{dist}{(1-dist)}}$$
$$sim(dist) = e^{-k\, dist^2}$$

donde k es un parámetro cuyo valor habrá que ajustar adecuadamente. Estas cuatro funciones cumplen las tres propiedades anteriores, y sus valores están normalizados entre 0 y 1. La gráfica de estas funciones se puede ver en la Fig. 1, en la que hemos representado los valores para $k = 1$ y $k = 20$ para cada una de ellas, estando los demás valores incluidos en el área delimitada por estos dos.

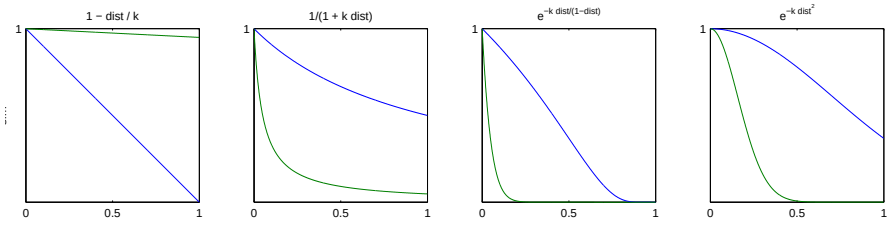

Figura 1. Funciones de similitud

Hemos aplicado una adaptación del algoritmo TextRank para calcular la importancia de la instancia, que nos ha permitido establecer un ranking entre las instancias, por el que las hemos ordenado. La expresión del cálculo del InstanceRank, donde el conjunto de entrenamiento es T y las instancias son $\{V_i, 1 \leq i \leq |T|\}$, es la siguiente:

$$IR(V_i) = (1 - d) + d \sum_{j=1}^{|T|} \frac{\text{sim}(V_j, V_i)}{\sum_{k=1}^{|T|} \text{sim}(V_j, V_k)} IR(V_j)$$

El ranking introducido por el InstanceRank dentro de las instancias se ha utilizado para su elección dentro del algoritmo WITS.

4.2. Bases de datos utilizadas, criterios de comparación y línea base

Las bases de datos con las que hemos trabajado han sido tomadas del Repositorio de Bases de Datos de Aprendizaje Automático de la Universidad de California en Irvine [11]. Las bases utilizadas son: Glass, Ionosphere, Iris, Pima, Sonar, Vote y Zoo, cuyas características se detallan en el Cuadro 1. Los clasificadores obtenidos con WIRS se han probado con el vecino más cercano (kNN con $k = 1$), utilizando validación cruzada estratificada con diez pliegues, por ser ésta la medida más extendida en el ámbito de la minería de datos y para poder comparar consistentemente los resultados con los de Morring y Martinez. Se ha programado dentro del entorno de WEKA [12] y utilizando su API, lo que facilita la escritura del clasificador y, especialmente, su evaluación.

Cuadro 1. Descripción de las bases de datos utilizadas

Base de Datos	Atributos Nominales	Atributos Reales	Número de Clases	% instancias clase mayoritaria	Número de instancias
Glass	0	10	7	35.51	214
Ionosphere	0	34	2	64.10	351
Iris	0	4	3	33.33	150
Pima	0	8	2	65.10	768
Sonar	0	60	2	53.37	208
Vote	16	0	2	61.38	435
Zoo	1	6	7	40.59	101

En la evaluación de los resultados se han considerado la precisión, medida en el porcentaje del número de aciertos sobre el total de ejemplos de test (siempre teniendo en cuenta que se usa validación cruzada con diez pliegues) y la reducción, medida en la proporción entre el número de instancias en el conjunto de instancias seleccionadas respecto al número de instancias en el conjunto de entrenamiento, de modo que un valor numérico más pequeño indica una mayor reducción.

Como línea base hemos tomado los resultados de la técnica del vecino más cercano sin ningún tipo de edición/reducción sobre los datos y los de WITS.

4.3. Ajuste de parámetros

En el desarrollo de WIRS se tuvieron que ajustar varios parámetros hasta conseguir los resultados óptimos. El parámetro G, que en WITS indica la mejoría en la clasificación a partir de la que una instancia se incluye en el conjunto se ha tomado igual: 0.005 veces el número de instancias del conjunto de entrenamiento, redondeando hacia arriba.

Los demás parámetros que hubo que ajustar fueron: la posible influencia de una instancia sobre sí misma en el cálculo del InstanceRank, el valor de d en éste, la función de similitud utilizada y el valor de k en estas funciones de similitud. Se comprobó que era preferible no considerar la influencia de una instancia sobre sí misma en el cálculo del rango. Para ajustar los otros parámetros se realizaron experimentos con un amplio rango de valores de k (entre 1 y 20) y d (entre 0.65 y 1.0, con intervalos de 0.05) para cada una de las funciones de similitud, estudiando en cada caso los valores de estos parámetros para los que se obtenían la mayor precisión media y la mayor reducción media en las siete bases de datos consideradas. También se estudió cuándo se obtenía el mejor compromiso entre reducción y precisión, siguiendo para ello el siguiente criterio: se ordenaron descendentemente los valores de k y d para los que se obtenía la mejor precisión, numerando cada uno de los casos; lo mismo se hizo para la reducción. Se tomó finalmente como valor de compromiso el que hacía que el producto del puesto en el ranking de reducción por el del puesto en el de precisión fuera menor. Los resultados se resumen en el Cuadro 2.

Los mejores resultados fueron consistentemente para la función $sim(d) = \frac{1}{1+k\,d}$, por lo que se pasó a estudiar ésta para valores de d con intervalos de 0.005. Se presentan los resultados en la Fig. 2, que se realizó con el fin de determinar la zona con mayor probabilidad de que haya un máximo; se observa que la zona de mayor precisión se da

Cuadro 2. Valores de k y d con reducción y precisión máximas, y mejor equilibrio, para cada una de las cuatro funciones de similitud

	reducción máxima				precisión máxima				equilibrio red/prec			
sim	red	prec	k	d	red	prec	k	d	red	prec	k	d
$1 - dist/k$	7.50	83.23	2	0.850	7.61	84.20	3	1.000	7.61	84.20	3	1.000
$1/(1 + k\,dist)$	7.47	84.73	9	0.955	7.71	85.83	8	0.900	7.70	85.57	8	0.940
$e^{-k\,dist/(1-dist)}$	6.90	84.13	4	1.000	7.62	84.52	2	0.900	7.55	84.48	4	0.850
e^{-kd^2}	7.15	84.47	6	0.850	7.72	84.88	13	0.900	7.68	84.52	12	0.900

alrededor de $k = 8$. En un análisis aún más detallado se determinó que los mejores valores en cuanto a equilibrio entre precisión y tasa de reducción se daban para $k = 8, d = 0{,}940$.

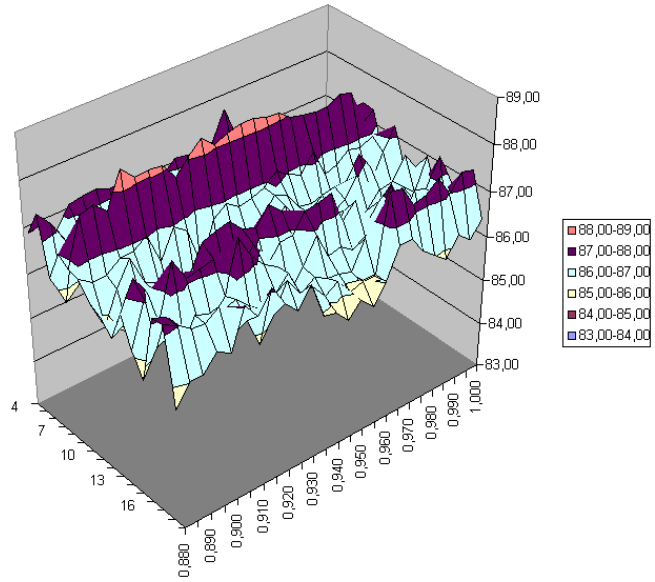

Figura 2. WIRS: precisión con $sim(dist) = \frac{1}{1+k\,dist}$, $4 \leq k \leq 20$, $0.875 \leq d \leq 1.000$

También se ajustó un parámetro propio de InstanceRank (y PageRank) que indica el nivel de error aceptable por debajo del cual se detienen las iteraciones. Se tomó un valor adecuado que consiguiera que el cálculo del InstanceRank se hiciera con la suficiente precisión como para garantizar la correcta ordenación de las instancias. Se estudió también el número de iteraciones necesarias para conseguir esa precisión observándose que se mantenía en unos niveles más que aceptables en cuanto a tiempo de ejecución (entre 3 y 10 iteraciones).

4.4. Resultados experimentales

Los resultados para estos valores de los parámetros, comparados con los obtenidos por kNN (con $k = 1$) y WITS para esos mismos conjuntos de datos se detallan en el Cuadro 3. Para cada técnica se detalla el tamaño del conjunto reducido respecto del original y la precisión. En el cuadro se han resaltado en negrita los casos en los que cada técnica ha destacado sobre la otra en tasa de reducción o en precisión. En los casos en los que la precisión mayor la consigue kNN, se ha resaltado en cursiva la técnica, de entre WITS y WIRS, que ha alcanzado mayor precisión. Naturalmente, no se produce ninguna reducción en kNN.

Cuadro 3. WIRS (con $sim(dist) = 1/(1 + k\,dist), k = 8, d = 0{,}94$), kNN y WITS: comparación en reducción y precisión

Base de datos	kNN ($k = 1$)		WITS		WIRS	
	% $\|S\|/\|T\|$	% Precisión	% $\|S\|/\|T\|$	% Precisión	% $\|S\|/\|T\|$	% Precisión
Glass	100.00	**73.83**	**9.03**	*65.79*	16.46	64.49
Ionosphere	100.00	84.62	7.33	88.06	**4.50**	**92.31**
Iris	100.00	94.00	**4.64**	93.53	6.07	**95.33**
Pima	100.00	73.56	1.56	74.48	**1.42**	**76.43**
Sonar	100.00	**87.55**	**7.58**	76.92	11.54	*79.81*
Vote	100.00	**95.64**	**1.36**	*94.83*	3.45	93.56
Zoo	100.00	94.44	**7.62**	94.06	10.45	**97.03**
Media	100.00	**86.23**	**5.59**	83.95	7.70	*85.57*

Como se observa, WIRS obtiene mayor reducción que WITS en dos de los conjuntos de datos (Ionosphere y Pima), en los que consigue además mayor precisión que WITS e incluso que kNN. WIRS consigue mayor precisión que WITS e incluso que kNN en otros dos conjuntos de datos, Iris y Zoo, aunque en éstos WITS es el que obtiene mayor reducción. En Vote WIRS queda ligeramente por debajo de WITS en precisión, aunque este último algoritmo es superado por kNN. En dos de las bases, Glass y Sonar, los resultados tanto de WITS como de WIRS son notoriamente peores que los de kNN; en uno de los casos WITS queda por encima (Glass), y en el otro WIRS (Sonar).

Resumiendo, WIRS resulta extraordinariamente competitivo en cuanto a precisión con uno de los algoritmos que actualmente marcan el estado del arte en técnicas de reducción de instancias. En algunos casos también resulta competitivo en cuanto a reducción.

5. Conclusiones

En este trabajo hemos presentado el algoritmo WIRS, una técnica de selección de instancias basada en el algoritmo WITS que utiliza la información proporcionada por un ranking de relevancia de instancias para determinar el orden en el que deben ser procesados los registros de la base de datos de entrenamiento. El ranking de relevancia de las instancias ha sido calculado con un algoritmo similar a PageRank, utilizado por Google para determinar la importancia de las páginas web de internet. Los resultados

View publication stats

de nuestros experimentos demuestran que esta aproximación es muy competitiva con respecto al algoritmo original WITS, especialmente en la precisión. En este aspecto conseguimos superar en un punto y medio (con un 85.57 de media en las siete bases de datos utilizadas) los resultados de WITS (83.95) quedando bastante cerca de la precisión media obtenida por la técnica de los vecinos más cercanos (86.23). También es interesante resaltar la capacidad de edición (o eliminación de instancias ruidosas) de nuestra técnica ya que en cuatro de las siete bases de datos estudiadas se mejoran los resultados de los vecinos más cercanos al reducir el número de instancias.

La principal conclusión de nuestro trabajo es que el hecho de interpretar la base de datos de entrenamiento mediante un grafo permite utilizar algoritmos de tratamiento de grafos (que como PageRank han demostrado ser eficaces en otros dominios) en el ámbito del aprendizaje automático. Como trabajos futuros inmediatos tenemos pensado experimentar sobre el algoritmo WIRS con algunas variantes para el cálculo del ranking, como la construcción de un grafo distinto para cada clase o la utilización de medidas de disimilitud además de las de similitud para incluir información negativa en el grafo.

Referencias

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine Learning **6** (1991) 37–66
2. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning **38** (2000) 257–286
3. Brighton, H., Mellish, C.: Advances in instance selection in instance-based learning algorithms. Data Mining and Knowledge Discovery **6**(2) (2002) 153–172
4. Papadimitriu, C.H., Bentley, J.L.: A worst-case analysis of nearest neighbor searching by projection. Lecture Notes in Computer Science **85** (1980) 470–482
5. Sproull, R.F.: Refinements to nearest-neighbor searching in k-dimensional trees. Algorithmica **6** (1991) 579–589
6. Morring, B.D., Martinez, T.R.: Weighted instance typicality search (WITS): A nearest neighbor data reduction algorithm. Intelligent Data Analysis **8**(1) (2004) 61–78
7. Zhang, J.: Selecting typical instances in instance-based learning. In: ML92: Proceedings of the ninth international workshop on Machine learning, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1992) 470–479
8. Wilson, D.R., Martinez, T.R.: Improved heterogeneous distance functions. Journal of Artificial Intelligence Research (JAIR) **1** (1997) 1–34
9. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems **30**(1–7) (1998) 107–117
10. Mihalcea, R., Tarau, P.: Textrank: Bringing order into texts. In Lin, D., Wu, D., eds.: Proceedings of EMNLP 2004, Barcelona, Spain, Association for Computational Linguistics (Julio 2004) 404–411
11. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases. (1998)
12. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann (2005)