

VNS para clasificación supervisada

Miguel García Torres, José A. Moreno Pérez, J. Marcos Moreno Vega

Resumen— El problema de la clasificación supervisada consiste en establecer la clase a la que asignar nuevos objetos descritos por una serie de atributos utilizando la información proporcionada por los atributos de objetos previamente clasificados. El objetivo consiste en obtener una regla que permita la clasificación de objetos descritos por una serie de atributos. El clasificador debe ser eficiente (consumir pocos recursos; tiempo y espacio) y eficaz (errar poco; en cuanto a número e importancia de los fallos). Un tipo de clasificador relevante es el de las reglas basadas en la comparación de los atributos de los nuevos objetos con los de un subconjunto de objetos, considerados representativos, previamente clasificados. La eficiencia de estos clasificadores depende del tamaño del subconjunto de objetos representativos. En este trabajo analizamos empíricamente la efectividad de la metaheurística de entorno variable (VNS) en la mejora de la eficacia del clasificador obtenido por métodos estándares manteniendo su eficiencia.

Palabras clave— Clasificación Supervisada, Variable Neighborhood Search

I. INTRODUCCIÓN

LA clasificación o clustering [13] es una técnica relevante del análisis exploratorio de datos. El interés de estas técnicas se justifica en la necesidad de disponer de procedimientos inteligentes que transformen en conocimiento la información contenida en enormes bases de datos. La minería de datos constituye uno de los campos de creciente importancia en el que la clasificación juega un papel central en la extracción de reglas de simplificación y en el descubrimiento de estructuras. Estas técnicas tienen también relevancia en otros campos de la ingeniería, en medicina y en las ciencias sociales.

La tarea de la clasificación se puede contemplar desde la perspectiva del aprendizaje supervisado o con un enfoque de aprendizaje no supervisado o de descubrimiento. En el contexto del aprendizaje no supervisado, no se conocen a priori las clases de los objetos disponibles. La tarea se concreta en la obtención de la clasificación óptima de un conjunto dado de objetos a partir de los atributos que los describen. La calidad de la clasificación se mide por el grado de similitud que existe entre los objetos que quedan encuadrados en una misma clase y la disimilitud entre los objetos asignados a clases distintas.

La clasificación no supervisada se realiza en dos

fases, en la primera se establecen la *clasificación* especificando las clases y en la segunda se determina un *clasificador* especificando la regla por la que se clasifica cada objeto. Los algoritmos de clasificación se engloban en dos tipos fundamentales en función de la estructura de clasificación que proporcionan: los algoritmos de clasificación jerárquica y los algoritmos de partición.

En la clasificación jerárquica se crea una secuencia de particiones anidadas donde los conjuntos que forman una partición están incluidos en los de la siguiente partición de la secuencia. Los algoritmos para obtener la clasificación jerárquica pueden ser de agregación o de separación. En los primeros se parte de un alto número de pequeñas clases que son gradualmente unidas en un número menor de clases mayores. En los algoritmos de separación se parte de un pequeño número de clases numerosas que se van dividiendo en un mayor número de clases más reducidas.

Los algoritmos de partición tratan de descubrir los agrupamientos presentes en los datos. Los objetos que se encuentran similares se agrupan en una misma clase y los que resultan muy distintos se colocan en clases diferentes.

Por otra parte, en el contexto del aprendizaje supervisado se parte de la existencia de un conjunto conocido de clases y se pretende obtener un clasificador que establezca una regla para asignar un objeto cualquiera a una de las clases a partir de los atributos que lo caracterizan. Se dispone de un conjunto de objetos de los que se conoce la clase a la que pertenecen y los atributos que los caracterizan. A este conjunto de objetos ya clasificados se le conoce como conjunto de entrenamiento y de él se pretende obtener una regla de clasificación o clasificador.

El clasificador se pretende aplicar a objetos del mismo tipo para conocer la clase en la que se encuadran. La calidad del clasificador se mide por su eficiencia (la cantidad de recursos que consume) y la eficacia (la garantía de que la asignación de la clase del objeto sea la correcta). Para medir el rendimiento del clasificador se utiliza un nuevo conjunto de objetos (conocido como conjunto de validación) del que se conoce la clase a la que pertenecen.

Existen distintos tipos de clasificadores en función de la forma en que utilizan la información disponible y el tipo de cómputo que realizan para

optar por una clase para cada objeto. En este trabajo consideramos clasificadores basados en el uso de la noción de distancia o similitud entre las descripciones de los objetos mediante la selección de un conjunto de descripciones representativas de las clases con las que realizar las comparaciones. Abordamos la optimización de clasificadores basados en la comparación de los nuevos objetos con objetos representativos de cada clase. Analizamos diversos aspectos de la aplicación de la metaheurísticas de búsqueda VNS para mejorar el conjunto de objetos de referencia que se toman como representantes de las clases. En particular, consideramos heurísticas constructivas usuales y búsquedas locales que se combinan en una búsqueda por entornos variables para ajustar de forma óptima el clasificador, una vez fijado por los métodos estándares los tamaños apropiados.

En la siguiente sección planteamos la formulación matemática del problema de la clasificación supervisada de objetos a partir de sus atributos. En la sección III describimos la metodología de clasificación basada en la similitud de los objetos a clasificar con las descripciones de referencia para cada clase. En la sección IV describimos los métodos constructivos que han sido propuestos y analizados para obtener el conjunto de referencia. En la sección V planteamos la obtención de un buen clasificador basado en descripciones de referencia como un problema de optimización combinatoria y describimos la aplicación de la metaheurística de arranque múltiple (VNS) para ajustar el clasificador. En la sección VI describimos la experiencia computacional llevada a cabo y finalizamos el trabajo con las conclusiones fundamentales en la sección VII.

II. CLASIFICACIÓN SUPERVISADA

El problema de la clasificación supervisada se plantea formalmente en los siguientes términos. Los *objetos* o casos a clasificar vienen descritos por medio de d *atributos* o variables X_j , $j = 1, 2, \dots, d$. La descripción del objeto x se expresa por los valores x_j de las variables X_j , $j = 1, 2, \dots, d$; por tanto denotamos $x = (x_1, x_2, \dots, x_d)$. Los atributos X_j , $j = 1, 2, \dots, d$, pueden ser cualitativos o numéricos de cualquier tipo con sus propios dominios, posiblemente diferentes en tamaño y tipo. Para aplicar los métodos aquí considerados sólo se precisa disponer de una forma de comparar los distintos valores de un mismo atributo y medir su similitud (o disimilitud).

Se dispone de las descripciones de un conjunto E de n objetos a_i , $i = 1, 2, \dots, n$, llamado conjun-

to de *entrenamiento*, por medio de los d atributos X_j , $j = 1, 2, \dots, d$. Por a_{ij} denotamos el valor del atributo X_j para el objeto a_i , $j = 1, 2, \dots, d$. Por tanto las descripciones del conjunto de entrenamiento vienen dadas por una matriz de datos $[a_{ij} : i = 1, 2, \dots, n; j = 1, 2, \dots, d]$.

Se dispone además de las descripciones, basadas en los mismos atributos, de otro conjunto V de m objetos b_i , $i = 1, 2, \dots, m$, llamado conjunto de *validación*. Equivalentemente denotamos por b_{ij} al valor del atributo X_j para el objeto b_i , $j = 1, 2, \dots, d$. Los datos disponibles del conjunto de validación también vienen dados por una matriz de datos $[b_{ij} : i = 1, 2, \dots, m; j = 1, 2, \dots, d]$.

Denotaremos los dominios de cada uno de los atributos X_j , $j = 1, 2, \dots, d$, por D_j . Por tanto las descripciones de los objetos son puntos o vectores del espacio d -dimensional $D_1 \times D_2 \times \dots \times D_d$. Además, se conoce la clase o categoría $c \in C$ a la que pertenece cada uno de los objetos, tanto del conjunto de entrenamiento como de validación. Ampliando la notación, cuando sea necesario supondremos que el atributo X_0 identifica la clase a la que pertenece el objeto correspondiente del dominio $C = D_0$. Por tanto a_{i0} y b_{i0} denotan respectivamente la clase del i -ésimo objeto del conjunto de entrenamiento o de validación.

El objetivo de la clasificación supervisada es utilizar la información proporcionada por el conjunto de entrenamiento para determinar un *clasificador* o regla de clasificación Cl que, dados los valores de los atributos X_1, \dots, X_d que describen un objeto x , asigne a cada objeto a una de las clases. Formalmente un clasificador es una aplicación del dominio D de las descripciones de los objetos en el dominio de las clases; $Cl : D \rightarrow D_0$.

La calidad de un clasificador se mide por su eficiencia y su eficacia. La *eficacia* del clasificador es el grado de coincidencia entre la clase asignada por el clasificador a los objetos a los que se le aplica y su verdadera clase. Además del número de fallos que se producen hay que tener en cuenta la magnitud de estos fallos. La *eficiencia* de un clasificador es la cantidad de recursos consumidos en su aplicación. Además del tiempo de cómputo hay que tener en cuenta la capacidad de memoria necesaria. Aparte de estos dos aspectos, hay que considerar la eficiencia del proceso de aprendizaje por el que se construye y actualiza el clasificador al incorporar nuevas descripciones al conjunto de aprendizaje.

III. DISTANCIAS

En este trabajo consideramos el diseño de clasificadores basados en las clases correspondientes a las descripciones más próximas a los objetos a

clasificar. Existen diversas formas de establecer comparaciones o medir el resultado de la comparación entre los valores de cualquier conjunto de atributos de distinto tipo. Estas alternativas se concretan en una distancia entre descripciones que consiste en una función de dos argumentos que es simétrica, no negativa y nula sólo si los valores de sus argumentos coinciden. Por tanto $Dist : D \times D \rightarrow \mathbb{R}^+$ es una distancia que debe verificar:

1. $Dist(x, x) = 0, \forall x \in D.$
2. $Dist(x, x') \geq 0, \forall x, x' \in D.$
3. $Dist(x, x') + Dist(x', x'') \geq Dist(x, x''), \forall x, x', x'' \in D.$

La distancia utilizada influye tanto en la eficiencia del clasificador como en su eficacia. Las distancias entre descripciones se suelen obtener a partir de distancias entre los valores de cada uno de los atributos. Estas distancias pueden ser diferentes aún cuando los atributos sean del mismo tipo y son posteriormente agregadas mediante un operador, que puede ser el máximo, pero que generalmente es la suma posiblemente ponderada. Las ponderaciones de las distancias de cada atributo se introducen para reflejar la distinta importancia de los atributos o para obtener algún tipo de normalización y reducir el efecto de la escala.

Sea $Dist_j$ la función de distancia, métrica o disimilitud entre los posibles valores de cada atributo X_j , para $j = 1, 2, \dots, d$. Si el atributo X_j es numérico se utiliza el valor absoluto de la diferencia de los valores como distancia; es decir,

$$Dist_j(x_j, x'_j) = |x_j - x'_j|, \forall x_j, x'_j \in D_j.$$

Para atributos no numéricos se han propuesto diversas distancias, desde las distancias dicotómicas hasta sofisticadas distancias que tratan de reflejar diferencias entre los valores más frecuentes de los atributos. La distancia dicotómica viene dada, $\forall x_j, x'_j \in D_j$, por:

$$Dist_j(x_j, x'_j) = \begin{cases} 1 & \text{si } x_j \neq x'_j \\ 0 & \text{si } x_j = x'_j \end{cases}$$

Si el atributo X_j es nominal o categórico una distancia alternativa más razonable es la métrica VDM (*value difference metric*) propuesta por [18] que trata de captar la distinta importancia de cada valor del atributo en el conjunto de entrenamiento. La métrica VDM entre dos valores x_j y x'_j del atributo X_j en referencia al conjunto de entrenamiento E viene dada, $\forall x_j, x'_j \in D_j$, por:

$$Dist_{VDM}(x_j, x'_j) = \sum_{c \in C} \left(\frac{N_j(x_j, c)}{N_j(x_j)} - \frac{N_j(x'_j, c)}{N_j(x'_j)} \right)^2.$$

Donde $N_j(x_j, c)$ es el número de veces que aparece el valor x_j en el atributo X_j de un objeto del conjunto de entrenamiento E en la clase c y $N_j(x_j) = \sum_{c \in C} N_j(x_j, c)$.

Dadas unas ponderaciones $w_j, j = 1, \dots, d$, que reflejan la distinta importancia de las distancias entre los atributos, la distancia o disimilitud entre las descripciones de dos objetos x y x' viene dada por:

$$Dist(x, x') = \sum_{j=1}^d w_j Dist_j(x_j, x'_j).$$

En [21] se encuentra una buena recopilación de esquemas de ponderación.

En el caso estándar de que todos los atributos sean numéricos, las descripciones de los objetos son puntos del espacio euclídeo d -dimensional \mathbb{R}^d . Las distancias usuales son las distancias de tipo ℓ_p , principalmente la distancia euclídea ℓ_2 , la distancia rectilínea ℓ_1 y la norma asintótica ℓ_∞ , además de diversas métricas o pseudométricas.

La distancia euclídea se define por:

$$Dist(x, x') = \|x - x'\|_2 = \sqrt{\sum_{j=1}^d (x_j - x'_j)^2}.$$

Si la distancia se utiliza para establecer qué descripciones están a mayor o menor distancia se puede prescindir de la raíz cuadrada. Se considera la distancia al cuadrado como función de disimilitud, en la que se pueden incluir las ponderaciones como factores de escala para obtener la fórmula

$$Dist_w(x, x') = \sum_{j=1}^d w_j (x_j - x'_j)^2.$$

También se puede usar cualquier otra norma ℓ_p ponderada, para cualquier $p \geq 0$. Esto da lugar a la función de disimilitud

$$Dist(x, x') = \sum_{j=1}^d w_j |x_j - x'_j|^p.$$

Especialmente relevantes, por la eficiencia de los cálculos son la norma ℓ_1 y la norma asintótica ℓ_∞ dadas, respectivamente por

$$Dist(x, x') = \|x - x'\|_1 = \sum_{j=1}^d w_j |x_j - x'_j|,$$

y por

$$Dist(x, x') = \|x - x'\|_\infty = \max_{j=1, \dots, d} w_j |x_j - x'_j|.$$

Por otro lado, puede ser conveniente utilizar las ponderaciones para realizar un proceso de normalización de los valores de los atributos para evitar, entre otras cosas, que un cambio de escala afecte al diseño y construcción del clasificador. En este proceso se pueden incluir, tanto atributos numéricos como nominales, e incluso datos perdidos, mediante la propuesta de la métrica de diferencia de valores heterogénea HVDM (*Heterogeneous Value Difference Metric*) de Wilson y Martínez [23] definida por

$$Dist_{HVDM}(x, x') = \sqrt{\sum_{j=1}^d w_j Dist_j(x_j, x'_j)}.$$

siendo:

1. Si alguno de los valores x_j o x'_j no existe entonces $w_j = 1$ y $Dist_j = 1$.
2. Si el atributo X_j es nominal entonces $w_j = 1$ y $Dist_j = Dist_{VDM}$.
3. Si el atributo X_j es numérico entonces $w_j = \frac{1}{16\sigma_j^2}$ donde σ_j^2 es la varianza de X_j en el conjunto de entrenamiento y $Dist_j = |\cdot|$.

La *eficacia* del clasificador se evalúa por el grado de coincidencia, para los objetos a los que se aplica, entre la clase asignada $Cl(x) = Cl(X_1, \dots, X_d)$ y la verdadera clase X_0 . Cualquiera de las distancias o formas alternativas de establecer la comparación entre atributos de los objetos, en este caso el atributo clase, especificadas anteriormente puede ser utilizado para evaluar la eficacia del clasificador sobre el conjunto de validación. Normalmente se utiliza el promedio de la distancia dicotómica, lo que se traduce en medir el éxito dado por el porcentaje o proporción de objetos del conjunto de validación correctamente clasificados. Si las clases tienen un significado semántico concreto, pueden existir clases más parecidas que otras, por lo que los errores en la aplicación de un clasificador pueden ser de distinto tipo y tener una valoración distinta utilizando un tipo de distancia diferente.

IV. DESCRIPCIONES DE REFERENCIA

Los métodos de clasificación basados en distancias o similitudes realizan medidas basadas en comparaciones de descripciones de los objetos a clasificar con la de los objetos, reales o hipotéticos, tomadas como referencia de cada clase. El criterio del vecino más próximo (NN: *Nearest Neighbour*) [4] implica asignar la clase del elemento de referencia que más se aproxima al objeto a clasificar.

Una primera alternativa es tomar un único objeto de referencia para cada clase. Los *centroides*,

obtenidos por los valores promedios de las descripciones de los objetos de cada una de las clases, constituyen una de las propuestas más comunes. Por tanto, una vez determinado los centroides $\{c_1, c_2, \dots, c_p\}$ de cada una de las p clases y se asigna cada nuevo objeto a la clase cuyo centroide le sea más próximo. El procedimiento de la J -medias propuesto por Hansen y Mladenovic [10] consiste en sustituir el centroide ficticio por un objeto de la clase.

En el lado opuesto de elegir sólo un representante de cada clase está la alternativa de tomar todos los objetos del conjunto de entrenamiento como referencia de las clases. Por tanto, aplicando el criterio del vecino más cercano, se asigna cada nuevo objeto a la clase a la que pertenece el objeto del conjunto de entrenamiento más próximo. Esta propuesta se generaliza, para paliar el efecto de los objetos mal clasificados, asignando a cada objeto la clase mayoritaria entre los k objetos más cercanos del conjunto de entrenamiento. Sin embargo, un valor de k mayor que 1 impide la consideración de clases con un único representante. Como valor de k se suele tomar un número impar pequeño; generalmente, 1, 3 o 5.

Una forma común de determinar un buen valor para k consiste en averiguar el valor de k que maximiza el número objetos del conjunto de entrenamiento en el que el k -ésimo objeto más cercano es de su misma clase, excluido el propio objeto. En algunos casos puede ser conveniente usar un valor $k > 1$ para construir el clasificador (obtener el conjunto de referencia) pero se suele dejar $k = 1$ para clasificar los objetos, o bien volver a determinar el valor más apropiado para k . En la exposición de este trabajo se toma $k = 1$ aunque los resultados se pueden extender casi directamente.

El clasificador del centroide es muy eficiente puesto que sólo tiene que hacer tantas comparaciones como clases y los requerimientos de espacio de memoria son muy bajos. El método del vecino más próximo pretende mejorar la eficacia pero a costa de un deterioro grave de la eficiencia en tiempo y espacio por almacenar todo el conjunto de entrenamiento y comparar el objeto a clasificar con todos sus elementos. Existen técnicas para reducir el tiempo necesario para determinar el vecino más cercano como las estructuras de árbol multidimensionales y las proyecciones. Para reducir los requerimientos de memoria se emplea el recurso de disminuir la dimensionalidad del problema seleccionando un subconjunto de atributos en los que basar las comparaciones. Esto también produce un ahorro de tiempo considerable, que junto con el ahorro de memoria

debe contraponerse a la pérdida de eficacia.

Como alternativa intermedia, entre elegir sólo un representante de cada clase y utilizar todo el conjunto de entrenamiento, se plantea la posibilidad seleccionar como objetos de referencia los elementos representativos del conjunto de entrenamiento. Por tanto, se asigna cada nuevo objeto a la clase a la que pertenece el objeto más próximo (o la mayoría de los k más próximos) entre los del conjunto de referencia $R \subseteq E$. Los elementos del conjunto de referencia se pueden obtener seleccionando objetos del conjunto de entrenamiento o diseñando descripciones ficticias, como los centroides, para representar conjuntos de objetos mediante prototipos [3], reglas [5], [6] o hiperrectángulos [20]. La selección de elementos del conjunto de entrenamiento para el conjunto de referencia sigue dos estrategias duales; algunos métodos tienden a elegir elementos centrales mientras que otros tienden a propiciar la selección de elementos fronterizos de las clases.

La reducción del conjunto de entrenamiento para obtener el conjunto de referencia seleccionando los elementos más representativos de cada clase mejora la eficiencia del clasificador al hacerlo más rápido y requerir menos espacio de memoria. Una adecuada selección, en tamaño y contenido, del conjunto de referencia debe permitir una mejora significativa de la eficiencia del clasificador sin deteriorar en exceso su eficacia. Un criterio demasiado complejo en la selección del conjunto de referencia provocaría un procedimiento de aprendizaje mediante su actualización demasiado costosa. Existen distintas alternativas para obtener el conjunto de referencia que proceden de forma constructiva o selectiva que desembocan en propuestas para el contenido y tamaño adecuado. Estos procesos se configuran como heurísticas constructivas que pueden ser complementadas con procesos eficientes de búsqueda local, búsquedas tabú o búsquedas de entorno variable.

Los métodos de obtención del conjunto de referencia son de dos tipos básicos: incrementales y decrementales. Los procedimientos *incrementales* parten de un conjunto de referencia vacío al que se van incorporando objetos del conjunto de entrenamiento mientras que los procedimientos *decrementales* parten del todo el conjunto de entrenamiento del que se van eliminando objetos. En ambos casos el proceso se detiene cuando se verifique algún criterio de parada que tenga en cuenta las características de la propuesta alcanzada. El orden en que se van considerando los objetos puede ser importante para ambos tipos de procedimiento. Se han propuesto versiones que

funcionan por lotes; en vez de seleccionar o eliminar los objetos uno a uno se determinan los que cumplen el criterio de selección o de eliminación y se insertan o eliminan todos ellos a la vez, repitiendo el proceso iterativamente [19].

Una ventaja de los métodos incrementales es que si se dispone de nuevos objetos para entrenamiento se puede continuar el aprendizaje aplicando el mismo criterio de selección. Además, puesto que sólo se usa la información de los objetos seleccionados, el proceso será muy rápido (al menos al principio) y requerirá menos memoria, sólo la necesaria para almacenar el conjunto de referencia final. Los procedimientos decrementales son más lentos y requieren más memoria pero, al trabajar desde el principio con más información, dan lugar a conjuntos de referencia más eficaces. Distintos métodos constructivos o de selección de un conjunto de referencia R (o un concepto análogo) para clasificación han sido estudiados en la literatura; ver [4], [1] y [24].

El método del *vecino más próximo condensada* (CNN) propuesto por Hart [12] busca como conjunto de referencia un subconjunto de R del conjunto de entrenamiento E tal que todo miembro de E esté más próximo a un elemento de R de su misma clase que otro miembro de R de clase distinta. El algoritmo puede empezar con un objeto por cada clase, como propone Hart, o con un conjunto vacío, como proponen Kibler y Aha [14]. En cada iteración, se considera la clasificación de todos los miembros de E por la clase del elemento de S más cercano y si un elemento de E queda mal clasificado se añade a R . Esta operación se repite hasta que todos los elementos de E queden bien clasificados por R . El método del *vecino más próximo selectivo* propuesta por Ritter et al. [17] busca un subconjunto R de E tal que todo miembro de E esté más próximo a un elemento de E (en vez de R) de su misma clase que otro miembro de R de clase distinta. El algoritmo es bastante complicado porque garantiza que el conjunto seleccionado es minimal con esta propiedad.

El método del *vecino más próximo reducido* (RNN) propuesto por [7] empieza con $R = E$ y se elimina de R todo aquel objeto que no da lugar a que otro objeto de E quede mal clasificado por los objetos que quedan en R . El algoritmo DROP1 de Wilson y Martínez [24] va eliminando de R todo aquel objeto que no da lugar a que otro objeto de R (en vez de E) quede mal clasificado por los objetos que quedan en R . El algoritmo de substracción propuesto por Kibler y Aha [14] también empieza con $R = E$ y elimina de R todo objeto que no quede mal clasificado por los

objetos que quedan en R .

El método del *vecino más próximo editado* (ENN) propuesto por Wilson [22] empieza con $R = E$ y se elimina de R todo aquel objeto que no discrepe de la mayoría de sus k vecinos más próximos en E . Este método fue modificado por Tomek [19] de la siguiente forma. Para i desde 1 hasta k marcar los objetos no clasificados correctamente por sus i vecinos más próximos. Una vez completado el bucle hasta k , eliminar los objetos marcados. Kubak y Matwin [15] aplican CNN y luego eliminan los objetos que intervienen en enlaces de Tomek; i.e., pares de objetos de clases distintas que se tienen por los más cercanos.

La métrica de similitud variable de Lowe [16] consiste en eliminar un objeto si los k vecinos más próximos son de la misma clase (aunque no coincida con el objeto a eliminar). Este método elimina los objetos mal clasificados.

V. MEJORA DEL CLASIFICADOR

En esta sección proponemos aplicar los métodos de búsqueda en optimización combinatoria para mejorar la eficacia del clasificador basado en un conjunto de referencia. La eficiencia del clasificador viene determinada por el tamaño del conjunto de referencia. Planteamos la búsqueda del mejor conjunto de referencia de un tamaño fijo como un problema de optimización combinatoria al que se le pueden aplicar metaheurísticas de búsqueda. En particular mostramos como aplicar la heurística VNS para mejorar las propuestas de los métodos clásicos anteriormente descritos.

Un problema de optimización combinatoria consiste en optimizar una función objetivo f en un espacio de soluciones S , siendo S un conjunto finito y f una función sobre S ; $f : S \rightarrow \mathbb{R}$. Una *solución óptima* s^* (o mínimo global) es una solución que optimiza la función f ; es decir, $s^* \in S$ tal que $f(s^*) \leq f(s)$, $\forall s \in S$. Una estructura de entornos \mathcal{N} es una aplicación de S en las partes de S ; es decir, $N(s) \subseteq S$, $\forall s \in S$. La solución $s' \in S$ es un *mínimo local* de f , con respecto a la estructura de entornos \mathcal{N} , si no existe una solución $s \in N(s') \subseteq S$ tal que $f(s) < f(s')$.

Para plantear la mejora del conjunto de referencia como un problema de optimización, las soluciones alternativas vienen dadas por las selecciones del conjunto de referencia. El objetivo del problema de optimización es la función que evalúa la eficacia del clasificador. Si R denota al conjunto de referencia, $R_j = R \cup C_j$ está formado por los objetos representativos de la clase C_j . Para clasificar un nuevo objeto x , se estima la distancia a cada clase C_j usando sólo los objetos

del conjunto de referencia mediante la fórmula

$$Dist(b_i, C_j) = \min_{a_j \in R_j} Dist(b_i, a_j).$$

Por tanto, para validar el clasificador que utiliza el conjunto de referencia R , cada objeto b_i del conjunto de validación se asigna a la clase $C_{j(i)}$ tal que:

$$Dist(b_i, C_{j(i)}) = \min_{j=1..p} Dist(b_i, C_j).$$

La clasificación obtenida viene determinada por la clase $C_{j(i)}$ asignada a cada uno de los objetos de validación b_i por medio de un vector o secuencia $j(i)$, $i = 1, 2, \dots, m$. Donde:

$$Cl(b_i) = j(i) = \arg \min_{j=1..p} Dist(b_i, C_j).$$

El clasificador es evaluado computando el porcentaje de éxitos en el conjunto de validación por:

$$f(R) = 100 \frac{|\{i : j(i) = b_{i0}\}|}{m}$$

Para resolver el problema de optimización necesitamos un algoritmo eficiente que permita encontrar, en el espacio de las clasificaciones posibles, la que minimice la función objetivo. El número de diferentes selecciones de q objetos representativos de un conjunto total de n objetos es del orden de n^q ; por ejemplo, para $n = 50$ y $q = 10$ es algo más de 10^{16} . Por tanto, no es apropiado utilizar un procedimiento de búsqueda que enumere todas las alternativas posibles, sino que, mediante alguna heurística, se debe seleccionar una buena clasificación estableciendo la partición y el correspondiente conjunto de centroides. Se trata de un problema de selección de tamaño fijo de los que abundan en optimización combinatoria.

Los procedimientos de búsquedas locales o búsquedas por entornos recorren el espacio de búsqueda por medio de una serie de movimientos empleando, por tanto, una estructura de entornos. Las búsquedas locales descendentes determinan en cada paso un movimiento que mejore la solución buscando en el entorno de la solución actual otra solución mejor, mientras exista. Las metaheurísticas de búsquedas por entornos tratan de utilizar estrategias inteligentes para continuar la búsqueda después de encontrar un mínimo local.

Muchas de las metaheurísticas de búsqueda están basadas en movimientos o transformaciones de la solución propuesta. El conjunto de movimientos considerados da lugar a una estructura de entornos en el espacio de búsqueda. El

entorno de cada solución está formado por aquellas soluciones a las que se puede acceder desde ella por uno de los movimientos considerados. Las características de la estructura de entornos son esenciales en el éxito de las estrategias de búsqueda aplicadas. Entre las diversas metaheurísticas de búsquedas por entornos que tratan de aprovechar de forma inteligente estas estructuras está la Búsqueda por Entorno Variable.

VI. METAHEURÍSTICA VNS

La Búsqueda por Entorno Variable (*Variable Neighborhood Search*, VNS) es una metaheurística propuesta sólo hace unos años que está basada en un principio simple: cambiar sistemáticamente la estructura de entorno en la búsqueda. Su desarrollo ha sido rápido, con varias docenas de artículos ya publicados a un ritmo creciente. Se han realizado muchas extensiones, principalmente para permitir la resolución de problemas de gran tamaño [8], [9], [11].

La estrategia de la VNS está basada en tres hechos muy simples:

1. Un mínimo local con respecto a una estructura de entornos no lo es necesariamente con otra.
2. Un mínimo global es un mínimo local con respecto a todas las posibles estructuras de entornos.
3. Para muchos problemas, los mínimos locales con respecto a una o varias estructuras de entornos \mathcal{N}_k están relativamente cerca unos de otros.

Esta última es una observación empírica que significa que un óptimo local frecuentemente proporciona información acerca del óptimo global. Este hecho se refleja, por ejemplo, en que varios aspectos de solución óptima local coincide con la óptima global; sin embargo, generalmente no se conoce en qué radica esa coincidencia. Parece entonces pertinente intentar un estudio del entorno de este óptimo local, hasta que se encuentre uno mejor.

En la mayoría de las aplicaciones de la VNS se ha hecho un esfuerzo en mantener la simplicidad del esquema básico. El esquema básico de una VNS es el siguiente. Sea \mathcal{N}_k , ($k = 1, \dots, k_{max}$), un conjunto finito de estructuras de entorno donde $\mathcal{N}_k(s)$ es el conjunto de soluciones del k -ésimo entorno de s . La mayoría de las heurísticas de búsqueda por entornos o búsquedas locales usan sólo una estructura de entornos. Los entornos \mathcal{N}_k pueden ser inducidos por una o más distancias en el espacio de soluciones alternativas.

El método de *búsqueda por entorno variable descendente* (VNDS; *Variable Neighborhood Descendent Search*) se obtiene si se aplica un cambio de

entornos de forma determinística en una búsqueda local descendente. Se selecciona un conjunto de estructuras de entornos \mathcal{N}_k , $k = 1, \dots, k_{max}$, que se usarán en el búsqueda. La descripción de la VNDS se muestra en la figura 1.

Inicialización. Obtener una solución inicial s ;
Repetir la siguiente secuencia hasta que no se obtenga mejora:

1. Hacer $k \leftarrow 1$;
2. Repetir los pasos siguientes hasta que $k = k_{max}$;

(a) Exploración del entorno. Encontrar la mejor solución s' del entorno de s ($s' \in \mathcal{N}_k(s)$);
(b) Moverse o no. Si la solución obtenida s' es mejor que s , hacer $s \leftarrow s'$ y $k \leftarrow 1$; en otro caso, hacer $k \leftarrow k + 1$.

Fig. 1. VNDS: VNS descendente

El método *reducido de búsqueda por entorno variable* (RVNS) se obtiene si simplemente se selecciona una solución al azar del entorno de la solución actual, $s' \in \mathcal{N}_k(s)$, sin realizar una búsqueda completa de la mejor solución del entorno. Generalmente se usa el máximo número de iteraciones entre dos mejoras como condición de parada en lugar de la inexistencia de mejora. La búsqueda por entorno variable reducida se presenta en la figura 2.

Inicialización. Obtener una solución inicial s ;
Repetir la siguiente secuencia hasta que no se cumpla el criterio de parada:

1. Hacer $k \leftarrow 1$;
2. Repetir los pasos siguientes hasta que $k = k_{max}$;

(a) Exploración del entorno. Encontrar una solución s' del entorno de s ($s' \in \mathcal{N}_k(s)$);
(b) Moverse o no. Si la solución obtenida s' es mejor que s , hacer $s \leftarrow s'$ y $k \leftarrow 1$; en otro caso, hacer $k \leftarrow k + 1$.

Fig. 2. RVNS; VNS reducida

La RVNS es útil para instancias muy grandes, para las que la búsqueda local completa sea muy costosa, si se usa alguna forma inteligente para encontrar la solución del entorno de la solución actual. Una posibilidad es generar al azar un número razonable de soluciones del entorno en función de los recursos disponibles.

El método *básico de búsqueda por entorno variable* (BVNS) combina cambios determinísticos y aleatorios de entorno. Su descripción se da en la figura 3.

La búsqueda local del paso (2b) puede ser sustituida por la VNDS.

Frecuentemente los sucesivos entornos \mathcal{N}_k ,

Inicialización. Obtener una solución inicial s ;
Repetir la siguiente secuencia hasta que no se cumpla el criterio de parada:

1. Hacer $k \leftarrow 1$;
2. Repetir los pasos siguientes hasta que $k = k_{max}$;

(a) Agitación. Generar al azar una solución s' del k -ésimo entorno de s ($s' \in \mathcal{N}_k(s)$);

(b) Búsqueda local. Aplicar algún método de búsqueda local con s' como solución inicial; denótese con s'' el mínimo local así obtenido.

(c) Moverse o no. Si la solución obtenida s'' es mejor que s , hacer $s \leftarrow s''$ y $k \leftarrow 1$; en otro caso, hacer $k \leftarrow k + 1$.

Fig. 3. BVNS: VNS básica

$k = 1, 2, \dots, k_{max}$, están anidados. Esto ocurre si sólo se dispone de una única estructura de entornos generada por un único tipo de movimientos que se aplica reiteradamente para producir una agitación de amplitud creciente. El método *anidado de búsqueda por entorno variable* (NVNS) utiliza un único conjunto de movimientos para producir entornos anidados en los que basar la búsqueda. Sea \mathcal{N} una única estructura de entornos. Se definen las estructuras de entornos anidados por $\mathcal{N}_1 = \mathcal{N}$, y

$$\mathcal{N}_{k+1}(s) = \bigcup_{s' \in \mathcal{N}(s)} \mathcal{N}_k(s'), (k = 1, \dots, k_{max} - 1).$$

La descripción de la búsqueda por entornos variable anidada se muestra en la figura 4.

Inicialización. Obtener una solución inicial s ;
Repetir la siguiente secuencia hasta que no se cumpla el criterio de parada:

1. Hacer $k \leftarrow 1$;
2. Repetir los pasos siguientes hasta que $k = k_{max}$;

(a) Agitación. Aplicar k veces un movimiento al azar desde la solución actual a una solución de su entorno. Sea s' la solución del k -ésimo entorno anidado de s ($s' \in \mathcal{N}_k(s)$);

(b) Búsqueda local. Aplicar la búsqueda local basada en la estructura de entornos \mathcal{N} con s' como solución inicial; denótese con s'' el mínimo local así obtenido.

(c) Moverse o no. Si la solución obtenida s'' es mejor que s , hacer $s \leftarrow s''$ y $k \leftarrow 1$; en otro caso, hacer $k \leftarrow k + 1$.

Fig. 4. NVNS: VNS anidada

La condición de parada aplicada en todas estas versiones suele ser el máximo tiempo de CPU permitido, máximo número de iteraciones, o máximo número de iteraciones entre dos mejoras.

VII. EXPERIENCIA COMPUTACIONAL

Para la experiencia computacional se han utilizado problemas obtenidos del repositorio de conjuntos de datos de la *UCI University of California (Irvine)* [2]. En particular se tomaron los problemas etiquetados *Iris*, *Wine*, *Bupa*, *Glass* y *Sonar*. Se trata de problemas clásicos en los que el número de casos u objetos y de variables o atributos viene reflejado en la tabla I. Debajo de la etiqueta de cada problema se indica entre paréntesis dichos valores, junto con el número de clases de cada uno de ellos. La distancia empleada fue, en todos los casos, la euclídea sin considerar normalización alguna.

Se implementaron tres técnicas para construir un conjunto de referencia que se relacionan a continuación y descritas anteriormente. 1) La técnica del vecino más próximo condensada (*CNN*) propuesta por Hart [12] pero realizada por lotes; se incorporan los objetos que verifican el criterio de inserción sin revisar el criterio por cada inserción. 2) La técnica del vecino más próximo reducida (*RNN*) propuesta por Gates [7]. 3) El algoritmo de substracción (*Shrink*) de reducción por el vecino más próximo sustractiva (*SNN*) propuesta por Kibler y Aha [14]. Además, a efectos comparativos, se aplicó el método del vecino más próximo con todo el de entrenamiento (*KNN*).

Siguiendo las pautas estándares para realizar las pruebas (ver [24]) se generaron 10 ordenaciones aleatorias de los objetos y se tomó el 90% del principio del banco de datos como conjunto de entrenamiento y el 10% final como conjunto de validación.

Se aplicó cada uno de los algoritmos *CNN*, *RNN* y *SNN* al conjunto de entrenamiento. Con cada resultado obtenido por estos algoritmos se aplicó la metaheurística VNS, partiendo de un conjunto R del mismo tamaño pero elegido al azar, (indicado en las tablas con la etiqueta *VNS*) y partiendo del conjunto R proporcionado por algoritmo (indicado por $+VNS$).

Para la implementación de la metaheurística VNS se adopta la versión anidada. Por tanto, como procedimiento de agitación se aplicó la sustitución de k objetos de R por k objetos de T . La selección de dichos objetos fue aleatoria. La agitación para un mismo valor de k se aplicaba mientras se obtuviera una solución mejor y si en 10 agitaciones no se mejora se pasa a $k + 1$. Los valores de k se incrementaban, mientras fuera necesario hasta $k_{max} = 10$. En el primer procedimiento *VNS* el conjunto R siempre aceptaba las nuevas incorporaciones a R introducidas por la agitación mientras que en el segundo sólo lo

TABLA I
COMPARACIÓN DE LA EFICACIA OBTENIDA EN EL
CONJUNTO DE ENTRENAMIENTO

Problema	Alg.	$ R $	NN	VNS	+VNS
Iris (150, 4) $c = 3$	CNN	10.69	98,32	97,92	99,01
	RNN	11.26	100,0	97,97	100,00
	SNN	8.96	91,56	98,07	98,42
Wine (178, 13) $c = 3$	CNN	35.12	93,84	84,80	94,38
	RNN	36.75	100,0	85,81	100,00
	SNN	34.56	95,87	84,72	96,86
Bupa (345, 6) $c = 2$	CNN	44.40	86,97	81,62	88,88
	RNN	52.23	100,00	84,71	100,00
	SNN	47.45	91,71	82,97	92,34
Glass (214, 10) $c = 7$	CNN	38.62	93,05	83,00	93,19
	RNN	41.82	100,00	84,84	100,00
	SNN	37.29	90,26	82,88	93,17
Sonar (208, 60) $c = 2$	CNN	29.72	93,75	87,69	94,14
	RNN	32.14	100,00	88,73	100,00
	SNN	26.79	90,59	86,54	92,91

aceptaba si mejoraba los resultados.

Para cada uno de los problemas, y para cada una de las 10 ordenaciones se ejecutó diez veces cada uno de estos algoritmos. Con cada conjunto de referencia obtenido se evaluó el porcentaje de aciertos tanto en el propio conjunto de entrenamiento como el de validación. Los resultados promedios se ofrecen respectivamente en las tablas I y II.

En la primera tabla (tabla I) se describen los indicadores de eficiencia obtenidos sobre el conjunto de entrenamiento. La primera columna ofrece información del problema: la etiqueta identificativa, el número de objetos y de atributos, y el número de clases. En la segunda columna etiquetada con $|R|$ se indica el porcentaje promedio de objetos que el algoritmo correspondiente incluyó en el conjunto de referencia. Las tres columnas siguientes indican el promedio del porcentaje de aciertos con los conjuntos de referencia obtenidos con cada algoritmo y con las dos formas de aplicar la metaheurística VNS. La tabla II ofrece el mismo tipo de información pero referentes al conjunto de validación.

Finalmente la tabla III indica los tiempos de ejecución, en segundos, correspondientes a cada una de las alternativas. Dichos algoritmos fueron ejecutados en un ordenador personal con procesador *Pentium III* a *1000 Mhz*. Los tiempos indicados como cero reflejan que el tiempo de cómputo es demasiado pequeño para poder ser medido.

TABLA II
COMPARACIÓN DE LA EFICACIA EN EL CONJUNTO DE
VALIDACIÓN

Problema	Alg.	$ R $	NN	VNS	+VNS
Iris (150, 4) $c = 3$	CNN	10.69	92,13	96,33	90,33
	RNN	11.26	93,33	96,07	85,00
	SNN	8.96	85,33	97,93	89,00
	KNN	100	98.67	–	–
Wine (178, 13) $c = 3$	CNN	35.12	73,11	74,89	73,33
	RNN	36.75	73,89	74,89	75,50
	SNN	34.56	70,56	75,72	71,00
	KNN	100	78.89	–	–
Bupa (345, 6) $c = 2$	CNN	44.40	57,60	59,51	59,71
	RNN	52.23	57,14	59,63	59,14
	SNN	47.45	54,00	60,00	56,74
	KNN	100	60.00	–	–
Glass (214, 10) $c = 7$	CNN	38.62	72,22	70,32	71,50
	RNN	41.82	72,72	69,50	69,91
	SNN	37.29	66,36	69,77	70,41
	KNN	100	75.91	–	–
Sonar (208, 60) $c = 2$	CNN	29.72	78,23	78,67	75,67
	RNN	32.14	79,04	79,71	73,95
	SNN	26.79	73,81	77,52	72,81
	KNN	100	83.33	–	–

TABLA III
COMPARACIÓN DE LA EFICIENCIA EN EL APRENDIZAJE

Problema	Alg.	NN	VNS	+VNS
Iris (150, 4) $c = 3$	CNN	1,0	0	0,5
	RNN	1,5	0	1,5
	SNN	0	0	0,5
Wine (178, 13) $c = 3$	CNN	1,0	1,5	2,5
	RNN	7,5	1,5	10,0
	SNN	0	1,5	2,5
Bupa (345, 6) $c = 2$	CNN	1	3,0	7,0
	RNN	31,0	3,5	37,0
	SNN	0	3,0	6,5
Glass (214, 10) $c = 7$	CNN	1,0	1,5	2,5
	RNN	11,0	1,5	13,5
	SNN	0	1,5	3,0
Sonar (208, 60) $c = 2$	CNN	3,0	6,5	12,0
	RNN	57,5	7,0	69,0
	SNN	0,5	6,0	12,0

VIII. CONCLUSIONES

Este trabajo muestra como las metaheurísticas son útiles para establecer el diseño de clasificadores basados en conjuntos de referencia, bien de forma independiente o para ajustar el clasificador obtenido por los métodos estándares de reducción de casos en aprendizaje supervisado. La metaheurística VNS utilizada se muestra más eficiente que los métodos estándares para obtener un ajuste máximo al conjunto de entrenamiento.

El principal inconveniente apreciado radica en la reducción de la eficacia obtenida sobre el conjunto de validación. Esto se debe a que el clasificador se ajusta demasiado al conjunto de entrenamiento. El desajuste que se produce con respecto al conjunto de validación se debe a las características del problema que no se ajustan al principio del vecino más próximo. Por ello se deben buscar alternativas al modelo de clasificador que tenga en cuenta esta particularidad.

Como líneas futuras de investigación se contempla: 1) el diseño y análisis de otro tipo de metaheurística, como las heurística de tipo GRASP, a estos problemas, 2) la aplicación de estas metaheurísticas al ajuste de otro tipo de clasificador más eficaz y eficiente y 3) la aplicación de estos procedimiento a la reducción de variables.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el Ministerio de Ciencia y Tecnología a través del proyecto TIC2002-04242-C03-01 con fondos que provienen, en un 70 %, del Fondo Europeo de Desarrollo Regional (FEDER).

REFERENCIAS

- [1] D.W. Aha, D. Kibler, M.K. Albert: Instance-Based Learning Algorithms. *Machine Learning* 6 (1991), pp. 37–66.
- [2] C.L. Blake, C.J. Merz: UCI Repository of machine learning databases, <http://www.ics.uci.edu/~mllearn/MLRepository.html>, University of California, Irvine, Dept. of Information and Computer Sciences (1998)
- [3] Chang, chin-Liang: Finding Prototypes for Nearest Neighbor Classifiers. *IEEE Transactions on Computers* 23-11, November 1974, pp. 1179–1184
- [4] B.V. Dasarathy: Nearest Neighbour (NN) norms: NN Pattern Classification Techniques. Los Alamitos, CA: IEEE Computer Society Press (1991)
- [5] P. Domingos: Rule Induction and instanced-Based Learning: A Unified Approach. in Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95), Montreal, Canada: Morgan Kaufmann, (1995) pp. 1226–1232.
- [6] P. Domingos: Unifying Instance-Based and Rule-Based Induction. *Machine Learning* 24 (1996), pp. 141–168
- [7] G.W. Gates: The Reduced Nearest Neighbor Rule. *IEEE Transactions on Information Theory* IT-18-3 (1972), pp. 431–433
- [8] P. Hansen, N. Mladenović: An Introduction to Variable Neighborhood Search. in S. Voss et al. (eds.). *Meta-Heuristics Advances and Trends in Local Search Paradigms for Optimization*. 1998, Kluwer.
- [9] P. Hansen, N. Mladenović: Variable Neighborhood Search Principles and Applications. *European Journal of Operational Research* 130, (2001) pp. 449–467.
- [10] P. Hansen, N. Mladenović: J-Means: A new local search heuristic for minimum sum of square clustering. *Pattern Recognition* 43(2), (2001) 401–413
- [11] P. Hansen, N. Mladenović: Developments of Variable Neighborhood Search. in C.C. Ribeiro and P. Hansen, *Essays and Surveys on Metaheuristics* Kluwer (2002).
- [12] P.E. Hart: The Condensed Nearest Neighbor Rule. *IEEE Transactions on Information Theory* 14 (1968), pp. 515–516.
- [13] J.A. Hartigan: Clustering Algorithms. (1975) John Wiley, New York.
- [14] D. Kibler, D.W. Aha: Learning Representative Exemplars of Concepts: An initial Case Study. Proceedings of the Fourth International Workshop on Machine Learning, Irvine, CA: Morgan Kaufmann (1987), pp. 24–30.
- [15] M. Kubat, S. Matwin: Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. In Fisher, D., ed., *Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97)*, Morgan Kaufmann Publishers, San Francisco, CA, (1997) pp. 179–186
- [16] D.G. Lowe: Similarity Metric Learning for a Variable-Kernel Classifier. *Neural Computation* 7-1, (1995) pp. 72–85
- [17] G.L. Ritter, H. B. Woodruff, S. R. Lowry, T.L. Isenhour: An Algorithm for a Selective Nearest Neighbor Decision Rule. *IEEE Transactions on Information Theory* 21-6, November 1975, pp. 665–669.
- [18] C. Stanfill, D. Waltz: Toward memory-based reasoning. *Communications of the ACM* 29, (1986) pp.1213-1228.
- [19] I. Tomek: An Experiment with the Edited Nearest-Neighbor Rule. *IEEE Transactions on Systems, Man, and Cybernetics* 6-6, (1976) pp.448–452
- [20] D. Wettschereck, T.G. Dietterich: An Experimental Comparison of Nearest-Neighbor and Nearest-Hyperrectangle Algorithms. *Machine Learning* 19-1, (1995) pp. 5–28.
- [21] D. Wettschereck, D.W. Aha, T. Mohri: A Review and Comparative Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review* 11, issues 1-5, Special Issue on Lazy Learning, Kluwer Academic Publishers, (1997) pp. 273–314.
- [22] D.L. Wilson: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics* 2-3, (1972) pp. 408–421.
- [23] D.R. Wilson, T.R. Martinez: Improved Heterogeneous Distance Functions. *Journal of Artificial Intelligence Research* 6-1, (1997) pp. 1–34.
- [24] D.R. Wilson, T.R. Martinez: Reduction Techniques for Instance-Based Learning Algorithms. *Machine Learning* 38-3, (2000) pp. 257–286.