# Aprendizaje Multi-instancia con Programación Genética para Web Mining

Amelia Zafra, Sebastián Ventura
Departamento de Informática
y Análisis Numérico
Universidad de Córdoba
14071 Córdoba
{azafra,sventura}@uco.es

Enrique Herrera-Viedma
Departamento de Inteligencia Artificial
y Ciencias de la Computación
Universidad de Granada
18071 Granada
viedma@decsai.ugr.es

**Resumen**

Este trabajo presenta un algoritmo de programación genética guiado por gramática para aprendizaje multi-instancia. Su funcionamiento es evaluado en el marco de Web Mining, con la finalidad de identificar páginas índice que contengan enlaces de interés para los usuarios. Esta nuevo algoritmo llamado G3P-MI es evaluado y comparado con otros algoritmos disponibles. Los experimentos computacionales muestran que el algoritmo G3P-MI obtiene resultados competitivos, resuelve el problema de otros algoritmos como la escalabilidad y añade comprensibilidad y claridad en el proceso de descubrimiento de conocimiento.

## 1. Introduction

El aprendizaje con multi-instancia (MIL) introducido por Dietterich et al. [4] es un marco de aprendizaje reciente que ha adquirido mucha importancia en la comunidad de machine learning. En este nuevo aprendizaje el conjunto de entrenamiento está compuesto por bolsas donde cada una contiene un conjunto de instancias. Una bolsa se etiqueta como positiva si contiene al menos una instancia positiva, en caso contrario se etiqueta como negativa. La información que se posee es relativa a las bolsas, no conociendo la clase a la que pertenece cada instancia individual. Como en todos los marcos de aprendizaje, la meta consiste en aprender conceptos a partir del conjunto de entrenamiento, en este caso para ser capaces de etiquetar correctamente nuevas bolsas.

Desde la aparición de MIL, siguiendo el trabajo de Dietterich et al. [4] han aparecido una gran cantidad de nuevos métodos. Auer [2] realizó una investigación teórica y presentó el algoritmo MULTINST. Un trabajo también teórico fue desarrollado por Long and Tan [7] que demostraron teóricamente que es posible la generación de un aprendizaje PAC desde ejemplos multi-instancia. Los primeros enfoques usando aprendizaje perezoso, árboles de decisión y reglas de aprendizaje fueron investigados durante el año 2000. En este contexto, encontramos varios trabajos, Whang and Zucker [12] propusieron dos variantes del algoritmo del vecino k más cercano (KNN) que ellos denominaron Citation-KNN y Bayesian-KNN. Con respecto a los árboles de decisión y las reglas de aprendizaje, Zucker and Chevaleyre [19] implementaron ID3-MI y RIPPER-MI. En la misma época, Ruffo [9] presentó una versión multi-instancia de árboles de decisión C4.5, que fue llamada RELIC. Más tarde, Zhou et al. [18] presentaron el algoritmo Fretcit-KNN, que es una variante del Citation-KNN. También podemos encontrar muchos otros algoritmos populares de machine learning adaptados al entorno MIL, tales como extensiones de las redes neuronales estándar [14, 15], máquinas de vector de soporte

[1, 16, 11] y el uso de ensembles [14].

De la misma forma, también han aparecido una gran variedad de tareas que se han formulado como problemas multi-instancias. Entre estas tareas, podemos encontrar la recomendación de páginas índice que consiste en determinar si una determinada página índice puede ser de interés para un usuario. Este problema ha sido resuelto tanto desde una perspectiva tradicional, con técnicas como el k vecino más cercano [10] o la frecuencia inversa del documento [5], como desde una perspectiva multi-instancia, adaptando el algoritmo del k vecino más cercano [18]. Sin embargo, las variantes del algoritmo del vecino más cercano presentan dos limitaciones principales. La primera está relacionada con la escalabilidad. Estos algoritmos no escalan bien cuando el número de items es elevado, viéndose afectada tanto la predicción como la precisión. El problema es que requieren cálculos que crecen linealmente con el número de items. La segunda, está relacionada con la interpretabilidad del conocimiento descubierto. Es decir, actúan como una caja negra, donde no se muestra ninguna información sobre las preferencias de los usuarios.

Aunque MIL ha sido investigado en una gran variedad de problemas, con una gran variedad de enfoques, no se ha abordado el problema con enfoques de algoritmos evolutivos [3] cuando estos algoritmos han demostrado excelentes resultados en otros marcos de aprendizaje. En este trabajo, pretendemos solucionar las limitaciones que presentan los métodos realizados hasta la fecha para la resolución del problema de recomendación de páginas índice proponiendo un nuevo algoritmo de MIL usando programación genética guidada por gramática (G3P) que nos permite descubrir las preferencias de los usuarios en la tarea de recomendación. Este nuevo algoritmo, llamado G3P-MI genera un clasificador basado en reglas sencillas que mejoran la habilidad de generalización, incluyen interpretabilidad y claridad en el conocimiento descubierto proporcionando información sobre los intereses de los usuarios y clasifican nuevos ejemplos (páginas webs) rápidamente. Resultados experimentales para resolver este problema muestran que este enfoque obtiene resultados competitivos en términos de *accuracy*, *recall* y *precision*.

El artículo se organiza en las siguientes secciones. La sección 2 describe el algoritmo G3P-MI propuesto. La sección 3 presenta el problema de recomendación de páginas índice. La sección 4 muestra los resultados experimentales. Finalmente, la sección 5 presenta las conclusiones y trabajo futuro.

## 2. Programación Genética Múltiple-instancia

La programación genética (GP) introducida por John Koza [6] se ha convertido en un paradigma con un creciente interés en tareas de clasificación. Podemos encontrar diferentes propuestas que muestran que GP es una campo con experiencia que logra errores bajos con respecto a otros algoritmos de aprendizaje supervisado [8, 13], con lo que a priori parece interesante su adaptación a este nuevo marco de aprendizaje.

Las siguientes secciones muestran las partes fundamentales de las que se compone nuestro sistema.

### 2.1. Representación de los Individuos

Un individuo está formado por dos componentes, *un genotipo* que es codificado con una estructura de árbol con profundidad limitada para evitar un tamaño demasiado grande, y *un fenotipo* que representa la regla completa formada por un antecedente y un consecuente. El antecedente consiste en la estructura de árbol y representa una regla que puede contener múltiples comparaciones unidas por conjunción o disyunción. El consecuente especifica la clase donde es clasificada la instancia que satisface todas las condiciones del antecedente. A partir de estos clasificadores, para llevar a cabo la clasificación de bolsas debemos modificar la definición formal de cubrimiento, y emplear la definición de cubrimiento multi-instancia dada por [17].

Para forzar las restricciones sintácticas y satisfacer la propiedad de cierre, se emplea

```
antecedente ->    comparacion
                | OR comparacion antecedente
                | AND comparacion antecedente

comparacion ->    opComparacion valoresComparacion

opComparacion ->    <
                  | ≥

valoresComparacion ->  atributo  valor
```

(a) Usando frecuencia de los términos

```
antecedente ->   comparacion
               | OR comparacion antecedente
               | AND comparacion antecedente

comparacion ->   opComparacion valoresComparacion

opComparacion ->    CONTIENE
                  | NO_CONTIENE

valoresComparacion ->  atributo
```

(b) No usando frecuencia de los términos

Figura 1: Gramática para la representación de los individuos

una gramática (ver Fig(1)). Esta gramática mantiene una restricción sintáctica y semántica tanto para la generación de individuos en la población inicial, como para la producción de nuevos individuos mediante los operadores genéticos.

### 2.2. Operadores Genéticos

Los elementos de la siguiente población son generados por medio de tres operadores: cruce, copia y elitismo.

#### 2.2.1. Cruce

El cruce se realiza intercambiando los subárboles de dos padres a partir de dos puntos compatibles seleccionados aleatoriamente en cada padre. Dos nodos árboles son compatibles si sus operadores pueden ser intercambiados sin producir un individuo inválido de acuerdo a la gramática definida. Si cualquiera de los descendientes es un individuo inválido, debido a la violación de la condición de unicidad para los atributos en cada antecedente de la regla o un árbol demasiado profundo, será reemplazados por uno de sus padres.

#### 2.2.2. Copia

El operador de copia simplemente elige un individuo en la población actual y lo copia sin cambios en la nueva población.

#### 2.2.3. Elitismo

El operador de elitismo copia el mejor individuo de una generación y lo pasa sin cambios a la siguiente generación, evitando así que el proceso estocástico de la evolución pierda a dicho individuo.

### 2.3. Función Fitness

La función de fitness evalúa la calidad de cada individuo de acuerdo a tres criterios básicos: *accuracy*, *recall* y *precision*. Para comprender estas medidas comentamos a continuación algunos conceptos necesarios, $P_a$ bolsas positivas que son recomendadas, $P_r$ bolsas positivas que son rechazadas, $N_a$ bolsas negativas que son recomendadas y $N_r$ bolsas negativas que son rechazadas. Siendo $P$ el número de bolsas positivas y $N$ el número de bolsas negativas, $P = P_a + P_r$ y $N = N_a + N_r$. La *accuracy*, *recall* y *precision* están definidas en las ecuaciones (1), (2) y (3), respectivamente:

$$accuracy = \frac{P_a + N_r}{P + N} \tag{1}$$

$$recall = \frac{P_a}{P} \tag{2}$$

$$precision = \frac{P_a}{P_a + N_a} \tag{3}$$

Nuestra función fitness es definida como el producto de todos los indicadores anteriores, ver (4). Todas las medidas son contrapuestas, conforme intentamos reducir el valor de alguna de ellas, las otras aumentarán, con lo que se intentará obtener una solución de compromiso entre ellas.

$$fitness = Accuracy \cdot Recall \cdot Precision \tag{4}$$

## 3. Recomendación de Páginas Índice como un Problema Multi-Instancia

Las páginas índices son páginas que contienen referencias o breves resúmenes de otras páginas. En Internet podemos encontrar un gran número de estas páginas. La recomendación de páginas índice [18], es una tarea clasificada dentro de las tareas de Web Usage Mining. Su finalidad es analizar las páginas índice automáticamente y mostrar al usuario sólo las páginas que contienen temás de su interés. Para ser capaces de decidir si una nueva página índice contiene algún tema de interés para el usuario, se necesita como paso previo identificar cuáles son sus intereses. La dificultad recae en que el usuario especifica si está interesado en páginas índice, en lugar de especificar los enlaces concretos en los que realmente está interesado.

Este problema puede ser visto como un problema multi-instancia donde la meta es etiquetar nuevas páginas índice como positivas o negativas. Una página índice es positiva si el usuario está interesado en al menos uno de sus enlaces. Una página índice es negativa si ninguno de sus enlaces es de interés para el usuario. En este problema, cada página índice será considerada como una bolsa mientras que sus páginas enlazadas serán consideradas como instancias en la bolsa. Cada instancia puede ser representada por un vector de términos $T = [t_1, t_2, , t_n]$, donde $t_i$ $(i = 1, 2, .., n)$ es uno de los $n$ términos más frecuentes que aparecen en la correspondiente página enlazada. Todas las páginas son descritas por el mismo número de términos, con lo que todas las instancias tendrán el mismo número de componentes, aunque éstos pueden ser muy diferentes. Para bolsas diferentes, sus correspondientes páginas índice pueden contener diferente número de enlaces, es decir, el número de instancias en las bolsas puede ser diferente.

## 4. Experimentos y Resultados

Llevaremos a cabo dos tipos de experimentos. Un primer experimento comparará el funcionamiento de nuestro algoritmo G3P-MI usando diferentes configuraciones sobre el problema de recomendación de páginas índice. El segundo experimento evalúa nuestro mejor algoritmo con otras técnicas de clasificación empleadas en la resolución de este problema. Todos los experimentos son repetidos cinco veces con diferentes semillas, y los valores medios de *accuracy*, *recall* y *precision* son mostrados en las siguientes secciones.

### 4.1. Datasets y Parámetros de Ejecución

Los experimentos han sido realizados sobre nueve conjuntos de datos, en cada uno de ellos un voluntario etiquetó 113 páginas índice de acuerdo a sus intereses. Para cada dataset, 75 páginas índice son seleccionadas aleatoriamente como bolsas de entrenamiento mientras las restantes 38 páginas índice son usadas como bolsas para test. Aunque mantemos exactamente las misma información establecida en [18], la conformación de los datasets es adaptada para trabajar con nuestro algoritmo. De esta forma se establece un vector por cada instancia, a su vez cada instancia contiene un atributo por cada palabra considerada y se le añaden dos atributos, el primero llamado $bolsa_{id}$ identifica la bolsa a la que pertenece y el segundo llamado $bolsa_{clase}$ determina la clase de su bolsa.

Todas las comparaciones de resultados son llevada a cabo por asociación de datasets. Los datasets son organizados en tres grupos, el primer grupo formado por los datasets V1, V2 y V3 contiene más bolsas negativas que positivas (este grupo es referenciado como conjunto de clase negativa mayoritaria). El segundo grupo está formado por los datasets V4, V5 y V6 que contienen más bolsas positivas que negativas (este grupo se referencia como conjunto mayoritario positivo) y el tercer grupo formado por los datasets V7, V8 y V9 contiene el mismo número de bolsas positivas que bolsas negativas (este grupo se referencia como conjunto balanceado). Esta asociación es interesante para darse cuenta y estudiar el comportamiento de los algoritmos entre datos balanceados y no balanceados.

Cuadro 1: Resumen de los resultados de G3P-MI

| | Mayoría de bolsas negativas | | | Mayoría de bolsas positivas | | | Conjunto Balanceado | | |
|---|---|---|---|---|---|---|---|---|---|
| Algoritmo | Acc | Rec | Prec | Acc | Rec | Prec | Acc | Rec | Prec |
| $G3P-MI^{a}$ | 0.842 | 0.468 | 0.369 | 0.807 | 1.000 | 0.801 | 0.693 | 0.927 | 0.621 |
| $G3P-MI^{b}$ | 0.807 | 0.690 | 0.379 | 0.825 | 0.877 | 0.895 | 0.711 | 0.750 | 0.727 |
| $G3P-MI^{1,a}$ | 0.842 | 0.559 | 0.660 | 0.816 | 1.000 | 0.809 | 0.474 | 1.000 | 0.474 |
| $G3P-MI^{1,b}$ | 0.886 | 0.821 | 0.518 | 0.825 | 1.000 | 0.816 | 0.526 | 1.000 | 0.500 |

1) Usando frecuencia de las palabras

a) Usando todas las palabras

b) Usando un conjunto de palabras más reducido

Los parámetros usados en todos las ejecuciones de GP fueron: tamaño de la población: 1000, generaciones: 100, probabilidad de cruce: 95 %, probabilidad de reproducción: 5 %, método de selección para ambos padres: la ruleta, profundidad máxima del árbol: 15. La población inicial fue generada usando el método ramped-half-and-half [6].

### 4.2. Comparaciones de G3P-MI

Se realizan dos comparaciones para evaluar el funcionamiento de nuestro algoritmo G3P-MI. La primera es entre datasets que consideran todas las palabras y datasets que consideran palabras que aparecen en más de 3 bolsas y menos de 40, este caso intenta eliminar tanto las palabras demasiado exclusivas como las demasiado usuales. Estas palabras no son útiles en tareas de clasificación e incrementan el espacio de búsqueda. La segunda es entre clasificadores que tienen en cuenta frecuencias de las palabras y clasificadores que sólo tienen en cuenta si las palabras aparecen o no aparecen. En la tabla 1 se muestra un resumen comparativo de los valores medios obtenidos.

Podemos ver que en caso de usar un dataset con un número reducido de palabras, además de reducir el espacio de búsqueda y requerir menos tiempo de cómputo, los resultados son mejores. Concretamente, los valores de *accuracy* y *precision* aparecen incrementados, mientras los valores de *recall* aparecen decrementados. Este comportamiento puede explicarse por el hecho de que el clasificador es generado para clasificar bolsas positivas, mientras que las bolsas que no cubren el clasificador, son consideradas bolsas negativas. Por ello, si eliminamos las palabras comunes y específicas, favorecemos que ciertas bolsas positivas no se incluyan porque no usamos palabras comunes que lo hacían incluirse (decrementamos el *recall*), pero a su vez cometemos menos errores, porque al eliminar las palabras comunes, menos bolsas negativas son consideradas como bolsas positivas (incrementamos la *precision*).

Si comparamos entre el método que considera la frecuencia de los términos o no, podemos ver que el primero obtiene mejores valores de *recall*, pero peores valores de *precision*. Esto nos indica que la inclusión de las frecuencias nos determina reglas más generales que clasifican la mayoría de las bolsas positivas a expensas de cometer errores incluyendo como positivas bolsas que no lo son. Esta relación que se produce entre estas dos medidas puede ser explicada si estudiamos el número de aciertos y errores producidos con respecto a cada clase. Un incremento de *recall* significa clasificar correctamente más bolsas de la clase positiva. De este modo, un incremento de verdaderos positivos (casos positivos correctamente identificados) está asociado con un incremento de falsos positivos (casos identificados como positivos pero realmente son negativos), y un

Cuadro 2: Resumen de los resultados generales

| | Mayoría de bolsas negativas | | | Mayoría de bolsas positivas | | | Conjunto Balanceado | | |
|---|---|---|---|---|---|---|---|---|---|
| Algoritmos | Acc | Rec | Prec | Acc | Rec | Prec | Acc | Rec | Prec |
| Fretcit-kNN | 0.879 | 0.622 | 0.514 | 0.854 | 0.925 | 0.896 | 0.698 | 0.603 | 0.718 |
| Txt-KNN | 0.795 | 0.652 | 0.335 | 0.805 | 0.985 | 0.809 | 0.570 | 0.722 | 0.538 |
| Citation-KNN | 0.803 | 0.433 | 0.336 | 0.796 | 0.864 | 0.875 | 0.674 | 0.561 | 0.701 |
| G3P-MI | 0.807 | 0.690 | 0.379 | 0.825 | 0.877 | 0.895 | 0.711 | 0.750 | 0.727 |
| Fretcit-kNN | 0.870 | 0.627 | 0.486 | 0.810 | 0.915 | 0.856 | 0.732 | 0.605 | 0.802 |
| Txt-KNN | 0.795 | 0.533 | 0.326 | 0.812 | 0.967 | 0.821 | 0.600 | 0.741 | 0.562 |
| Citation-KNN | 0.833 | 0.456 | 0.429 | 0.782 | 0.852 | 0.858 | 0.674 | 0.594 | 0.695 |
| G3P-MI | 0.886 | 0.821 | 0.518 | 0.825 | 1.000 | 0.816 | 0.526 | 1.000 | 0.500 |

decremento de los verdaderos negativos (casos negativos identificados correctamente). Por lo tanto, cualquier incremento en *recall* suele ir acompañado por un decremento de la *precision*.

### 4.3. Comparición con Otros Algoritmos

Nuestro algoritmo es comparado con otros existentes, que son Frecit-KNN, Citation-KNN y Txt-KNN. Ellos son descritos en [18]. Txt-KNN es obtenido adaptando el algoritmo KNN estandar a los objetos textuales. Citation-KNN es similar al anterior pero considera tanto referencias como citas de los objetos. Frecit-KNN es similar a Citation-KNN, pero a diferencia de los anteriores, este es un algoritmo de múltiple instancia. Después de analizar nuestro algoritmo en secciones anteriores, elegimos para la comparación la configuración que reducía el número de palabras consideradas. Los resultados de los algoritmos se muestran en la tabla 2.

Si comparamos todos los métodos, podemos ver que G3P-MI obtiene mejores resultados que Txt-KNN y Citation-KNN. Con respecto a Frecit-KNN, en caso de no considerar las frecuencias de palabras, nuestro algoritmo obtiene resultados ligeramente mejores para datos balanceados, aunque son ligeramente peores para datos no balanceados. En nuestra opinión, la gran ventaja que presenta nuestro algoritmo es la habilidad de generar reglas inteligibles que una vez interpretadas proporcionan una información muy valiosa para conocer los intereses del usuario. A continuación, se muestra un clasificador obtenido para el dataset V1.

**SI** (*(contiene* ***planeta*** *Y* ***deforestación****) O*
*(contiene* ***atmósfera****)*
*O (no_ contiene* ***fútbol****)* **)**
**ENTONCES**
*Recomendar la página al usuario V1.*
**SINO**
*No recomendar la página al usuario V1.*

Del clasificador generado para el usuario V1, podemos aprender temas que le interesan al usuario, conociendo sus preferencias. Así, estaría interesado en temas como ecología y entorno (con palabras de interés como planeta, deforestación y atmósfera) y no está interesado en temas deportivos relativos al fútbol.

Finalmente, podemos resumir dos motivaciones principales para usar G3P-MI:

1. No es necesario establecer el número de términos considerados en cada instancia para la ejecución del algoritmo. Como es conocido, el algoritmo del vecino k más

cercano con el que se compara, es una técnica costosa que requiere cómputos que crecen linealmente con el número de términos.

2. Los algoritmos disponibles para resolver este problema no generan hipótesis interpretables tales como conjunto de reglas. Nuestro algoritmo genera conocimiento simple, modular e intuitivo. Además, el conjunto de reglas inducido por G3P-MI es simple (sólo contiene unos cinco o seis literales), facilitando de este modo la comprensibilidad.

## 5. Conclusiones

Este artículo describe un primer intento de aplicar técnicas G3P para MIL. Es comparado con las tres técnicas más empleadas para resolver el problema de recomendación de páginas índices, Txt-KNN, Citation-KNN y Frecit-KNN. G3P-MI obtiene resultados competitivos en términos de *accuracy*, *recall* y *precision*, sus resultados son significativamente mejores que dos técnicas y con respecto a la última mencionada son ligeramente mejores para datos balanceados, y ligeramente peores en los no balanceados. Sin embargo, la característica más relevante de G3P-MI es que añade los beneficios de la interpretabilidad y la claridad proporcionando información sobre las preferencias de los usuarios, mientras que los otros algoritmos no son capaces de generar conocimiento fácilmente comprensible.

El problema de recomendación de páginas índice para algoritmos tales como G3P es complejo, debido al gran número de atributos, lo que causa que algunas hipótesis no sean predictivas debido a que el espacio de búsqueda es demasiado amplio. Actualmente, estamos investigando algoritmos de selección de características para solucionar este problema y así decrementar el espacio de búsqueda y mejorar los resultados.


## Agradecimientos

Este trabajo ha sido financiado en parte por el proyecto TIN2005-08386-C05-02 de la comisión inter-ministerial de ciencia y tecnología (CICYT), fundación FEDER y proyecto SAINFOWEB P05-TIC-602.


## Referencias


[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS'02: Proceedings of Neural Information Processing System*, pages 561–568, 2002.

[2] P. Auer. On learning from multi-instance examples: Empirical evaluation of a theoretical approach. In *ICML'97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 21–29, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers.

[3] T. Back. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, Oxford, UK, 1996.

[4] T.G. Dietterich, R.H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artifical Intelligence*, 89(1-2):31–71, 1997.

[5] T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *ICML'97: Proceedings of 14th International Conference on Machine Learning)*, pages 143–151, Nashville, US, 1997. Morgan Kaufmann Publishers.

[6] J.R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Dic 1992.

[7] P.M. Long and L. Tan. PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance

examples. *Machine Learning*, 30(1):7–21, 1998.

[8] D.P. Muni, N.R. Pal, and J. Das. A novel approach to design classifiers using genetic programming. *IEEE Transactions Evolutionary Computation*, 8(2):183–196, 2004.

[9] G. Ruffo. *Learning single and multiple instance decision tree for computer security applications*. PhD thesis, Department of Computer Science. University of Turin, Torino, Italy, 2000.

[10] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing)*. The MIT Press, 2006.

[11] Q. Tao, S. Scott, N.V. Vinodchandran, and T. Takeo Osugi. SVM-based generalized multiple-instance learning via approximate box counting. In *ICML'04: Proceedings of the twenty-first international conference on Machine learning*, pages 799–806, New York, NY, USA, 2004. ACM Press.

[12] J. Wang and J. Zucker. Solving the multiple-instance problem: A lazy learning approach. In *ICML'00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1119–1126, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers.

[13] M. Zhang and W. Smart. Using gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. *Pattern Recognition Letters*, 27(11):1266–1274, August 2006.

[14] M. Zhang and Z. Zhou. Ensembles of multi-instance neural networks. In *Intelligent information processing II*, volume 163, pages 471–474, London, UK, 2005. Springer Boston.

[15] M. Zhang and Z. Zhou. Adapting rbf neural networks to multi-instance learning. *Neural Processing Letters*, 23(1):1–26, 2006.

[16] Q. Zhang and S. Goldman. EM-DD: An improved multiple-instance learning technique. In *NIPS'01: Proceedings of Neural Information Processing System 14*, pages 1073–1080, 2001.

[17] Z. Zhou. Multi-instance learning from supervised view. *Journal Computer Science and Technology*, 21(5):800–809, 2006.

[18] Z. Zhou, K. Jiang, and M. Li. Multi-instance learning based web mining. *Applied Intelligence*, 22(2):135–147, 2005.

[19] J. Zucker and Y. Chevaleyre. Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. Application to the mutagenesis problem. In *Proceedings of the 14th Canadian Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence*, pages 204–214, Ottawa, Canada, 2000.