Diseño Automático de Multi-Clasificadores Basados en Proyecciones de Etiquetas

Jose M. Moyano, Eva L. Gibaja, Alberto Cano, Jose M. Luna, y Sebastián Ventura

Universidad de Córdoba. Departamento de Informática y Análisis Numérico

Resumen El aprendizaje multi-etiqueta es una tarea de aprendizaje supervisado que trata con problemas en los que un patrón puede tener asociadas a la vez varias clases binarias. Se trata de un paradigma en auge debido a la cantidad de problemas reales que se pueden abordar y su complejidad, fundamentalmente a causa de la alta dimensionalidad en el espacio de salida. En este trabajo se pretende abordar esta gran dimensionalidad aplicando un enfoque de *ensembles* en el que, mediante proyecciones del espacio de salida, cada clasificador base tiene en cuenta únicamente un subconjunto de etiquetas. Esta propuesta, llamada EME (*Evolutionary Multi-label Ensemble*), se evalúa y compara con otros algoritmos del estado del arte, obteniendo un rendimiento consistentemente mejor que muchos de los algoritmos de referencia.

Keywords: Aprendizaje multi-etiqueta, ensembles, algoritmos genéticos

1. Introducción

Existen multitud de problemas del mundo real donde los objetos pueden asociarse con más de una clase simultáneamente. Para ello, el aprendizaje multi-etiqueta (*Multi-Label Learning*, MLL) elimina la restricción clásica de que un patrón pueda tener asociada una única clase. Este tipo de aprendizaje ha sido aplicado con éxito en campos como categorización de textos [7], minería de redes sociales [13] o biología [2]. En MLL cada instancia de un conjunto de datos d-dimensional $\mathcal{X} = X_1 \times \cdots \times X_d$, podrá tener asociado un conjunto de una o varias etiquetas $Y \subseteq \mathcal{L} = \{\lambda_1, \ldots, \lambda_q\}, q > 1$. Generalmente, los problemas de aprendizaje multi-etiqueta tienen una alta dimensionalidad en el espacio de salida, por lo que los modelos pueden llegar a ser costosos en tiempo de predicción o en memoria. Una de las soluciones propuestas para tratar con este tipo de problemas es hacer uso de *ensembles* o multi-clasificadores, donde se combinan varios clasificadores base, tratando de que el *ensemble* completo tenga mejores resultados que los clasificadores base por separado.

Hasta el momento, se han desarrollado varios modelos para aprendizaje multi-etiqueta basados en *ensembles* [5]. El algoritmo propuesto en este trabajo, llamado EME (*Evolutionary Multi-label Ensemble*), diseña automáticamente *ensembles* basados en proyecciones de etiquetas mediante algoritmos evolutivos.

Cada uno de los clasificadores base tiene en cuenta un subconjunto de las etiquetas de tamaño fijo, y la decisión final del *ensemble* se obtendrá por votación. De este modo, se tiene en cuenta la relación existente entre etiquetas con un coste computacional relativamente bajo, optimizándose tanto la combinación de etiquetas en cada uno de los clasificadores base como los clasificadores base que formarán parte del *ensemble*. Además, la función de *fitness* tiene en cuenta tanto la exactitud del algoritmo como que todas las etiquetas se consideren un mismo número de veces en el *ensemble*. El rendimiento ha sido evaluado y comparado con otros algoritmos del estado del arte sobre conjuntos de datos de diversos dominios.

El resto del artículo se organiza de la siguiente manera: la Sección 2 incluye conocimientos del trabajo existente en aprendizaje multi-etiqueta, la Sección 3 presenta el algoritmo evolutivo, la Sección 4 presenta los experimentos diseñados, la Sección 5 muestra los resultados obtenidos y su discusión, y por último, la Sección 6 muestra las conclusiones y algunas líneas de trabajo futuro.

2. Antecedentes

En esta sección se describen distintos modelos existentes en clasificación multi-etiqueta y se introducen las métricas de evaluación específicas de este tipo de aprendizaje.

2.1. Algoritmos de Clasificación Multi-Etiqueta

En aprendizaje multi-etiqueta, existen dos grandes tipos de algoritmos: los métodos de transformación de problemas y la adaptación de algoritmos [5]. Los métodos de transformación de problemas convierten un problema multi-etiqueta en uno o varios problemas multi-clase, que son resueltos por algoritmos de clasificación tradicionales. Uno de los más populares es *Binary Relevance* (BR) [16], que genera un clasificador binario para cada etiqueta, tratándolas de manera independiente. *Label Powerset* (LP) [3] transforma el problema multi-etiqueta en un problema multi-clase, creando una nueva clase por cada combinación de etiquetas distinta que aparece en el *dataset*. *Pruned Sets* (PS) [9] trata de reducir la complejidad de LP, podando los patrones que tengan combinaciones de etiquetas poco frecuentes. Y por último, *Classifier Chain* (CC) [11] genera q clasificadores binarios encadenados, donde cada clasificador incorpora como atributos adicionales las etiquetas predichas en los clasificadores anteriores en la cadena.

Se habla de *ensembles* de clasificadores multi-etiqueta cuando los clasificadores base que los forman son también clasificadores multi-etiqueta [8]. RAkEL (*RAndom k-labELsets*) [17] construye un *ensemble* de clasificadores LP, donde cada uno entrena un subconjunto aleatorio de etiquetas, y la decisión final viene dada por votación de los clasificadores base. Por otra parte, *Ensemble of Pruned Sets* (EPS) [9] crea un *ensemble* de PSs donde cada clasificador está entrenado con un muestreo del conjunto de entrenamiento sin remplazamiento. *Ensemble*

of Classifier Chains (ECC) [1] entrena un *ensemble* de CCs, cada uno con un encadenamiento aleatorio y una muestra con reemplazamiento del conjunto de entrenamiento. *Multi-Label Stacking* (MLS) [14], también llamado 2BR, consiste básicamente en aplicar BR dos veces. Durante el primer paso (nivel base), se entrena un clasificador BR, y en el segundo (meta-nivel) se genera un nuevo clasificador cuyas entradas son las salidas de los clasificadores anteriores. Para finalizar, HOMER (*Hierarchy Of Multi-label classifiERs*) [15] genera un árbol de tareas MLL, donde cada una tiene un pequeño número de etiquetas. En cada nodo, las etiquetas se dividen con un algoritmo de *clustering*, agrupando etiquetas similares en una meta-etiqueta. Las hojas representan las etiquetas del *dataset*.

2.2. Métricas de Evaluación

En aprendizaje multi-etiqueta, la predicción para un patrón puede ser totalmente correcta, solo de manera parcial o completamente incorrecta, dependiendo si se predicen correctamente todas las etiquetas, solo algunas, o ninguna respectivamente. Por tanto, existe la necesidad de definir métricas de evaluación específicas, que se dividen en métricas basadas en etiquetas y métricas basadas en ejemplos.

Métricas Basadas en Etiquetas Cualquier métrica de evaluación para clasificación binaria se puede calcular por la aproximación basada en etiquetas (ej. *precision*, *recall* o *specificity*). La idea es calcularlas en base a los valores de la matriz de confusión (*true positives*, *false positives*, *false negatives* y *true negatives*) [16]. Dada una métrica de evaluación binaria, la aproximación micro primero une todas las matrices de confusión y después calcula el valor de la métrica, y la aproximación macro calcula el valor de la métrica para cada etiqueta y después promedia entre el número de etiquetas [18].

Métricas Basadas en Ejemplos Las métricas basadas en ejemplos calculan el valor de la métrica para cada instancia, y luego promedia entre el número de instancias total, así tiene en cuenta la relación entre las etiquetas. El *Hamming loss* [12] evalúa cuantas veces, en media, una etiqueta no se predice correctamente. Tiene en cuenta tanto los errores de predicción (una etiqueta negativa se predice como positiva) como los errores de omisión (una etiqueta positiva no se predice). *Subset accuracy* [19] indica la fracción de instancias cuyas etiquetas predichas son exactamente las etiquetas reales. Es una métrica muy estricta, porque requiere que coincidan todas las etiquetas predichas con las reales. También son definidas con este enfoque otras métricas de clasificación clásicas como *precision*, *recall*, *F-Measure*, *specificity* o *accuracy* [6].

3. Algoritmo

En esta sección se presenta el algoritmo EME, centrándose en el esquema de codificación, los operadores genéticos, y la función de *fitness* aplicada en el algoritmo evolutivo.

3.1. Individuos

Cada individuo de la población representa un *ensemble* multi-etiqueta. El genotipo de cada individuo es un vector binario de tamaño $n \times q$, siendo n el número de clasificadores base del *ensemble* y q el número de etiquetas. Los *ensembles* están basados en proyecciones del espacio de salida, donde cada clasificador base se centrará en un subconjunto de las etiquetas del conjunto total, representado con k bits a 1 en cada clasificador base. En el ejemplo de la Figura 1, cada fila representa un clasificador base, donde, por ejemplo, el clasificador MLL_1 clasificaría las etiquetas λ_1, λ_3 y λ_4. Todos los clasificadores base del *ensemble* son del mismo tipo, diferenciándose en la proyección de etiquetas. La población inicial se genera asignando k bits aleatorios a 1 en cada clasificador base.

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
MLL_1	1	0	1	1	0	0
MLL_2	1	1	0	0	0	1
MLL_3	0	0	1	0	1	1
MLL_4	0	1	0	1	1	0
MLL_5	1	0	0	1	0	1
MLL_6	0	0	1	1	1	0
MLL_7	0	1	1	0	0	1
MLL_8	1	0	0	0	1	1

Figura 1. Genotipo del individuo

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
MLL_1	0	–	1	0	–	–
MLL_2	1	0	–	–	–	1
MLL_3	–	–	0	–	1	1
MLL_4	–	0	–	0	1	–
MLL_5	1	–	–	1	–	1
MLL_6	–	–	1	0	1	–
MLL_7	–	0	0	–	–	0
MLL_8	1	–	–	–	1	0
	3/4	0/3	2/4	1/4	4/4	3/5
Umbral: 0.5	1	0	1	0	1	1

Figura 2. Proceso de votación

Antes de evaluar los individuos hay que generar los *ensembles* que representan, filtrando para cada clasificador base únicamente las etiquetas que ha de tener en cuenta. Un ejemplo del proceso de votación del *ensemble* se muestra en la Figura 2. En este ejemplo, la etiqueta λ_1 obtiene 3 votos positivos de 4 posibles, por lo que la predicción final es positiva, ya que supera el ratio de votos del umbral (0.5), mientras que para λ_2 todos los votos recibidos son negativos, así que la predicción final será negativa.

3.2. Función de Fitness

Una vez que se genera el *ensemble*, se calcula el valor de la función de *fitness*. El *fitness* mide por un lado el rendimiento del clasificador y por otro que todas las etiquetas se tengan en cuenta el mismo número de veces en el *ensemble*, evitando así que haya etiquetas que estén presentes en la mayoría de clasificadores base y otras que se dejen de lado.

La métrica escogida para medir el rendimiento del clasificador es el *Hamming loss*, que mide el error en las predicciones. Para asegurar que el número de votos por etiqueta es homogéneo se define una métrica de cobertura. La cobertura se define como la distancia entre el número de votos esperado para cada etiqueta y el número de votos real en el *ensemble*, siendo $votosEsperados = \frac{k \cdot n}{q}$ (votos totales entre número de etiquetas).

$$cobertura = \frac{\sqrt{\sum_{i=1}^{q}(votosEsperados - votos[i])^2}}{q} \quad (1)$$

Por tanto, el *fitness* final será una combinación lineal de estas dos métricas, *Hamming loss* y cobertura, ambas a minimizar:

$$\downarrow fitness = HLoss + cobertura \quad (2)$$

3.3. Operadores Genéticos

Operador de Cruce Para cada individuo de la población se decide, en función de una probabilidad de cruce, si pasa a formar parte del conjunto de padres. Después, se escogen pares de individuos aleatorios de este conjunto para aplicar el cruce. Para cada uno de los clasificadores base del *ensemble* el operador decide en base a una probabilidad si se intercambia con el clasificador base que ocupa la misma posición en el otro padre (Figura 3). Los nuevos individuos serán siempre individuos factibles, ya que no se modifica el número de etiquetas activas.

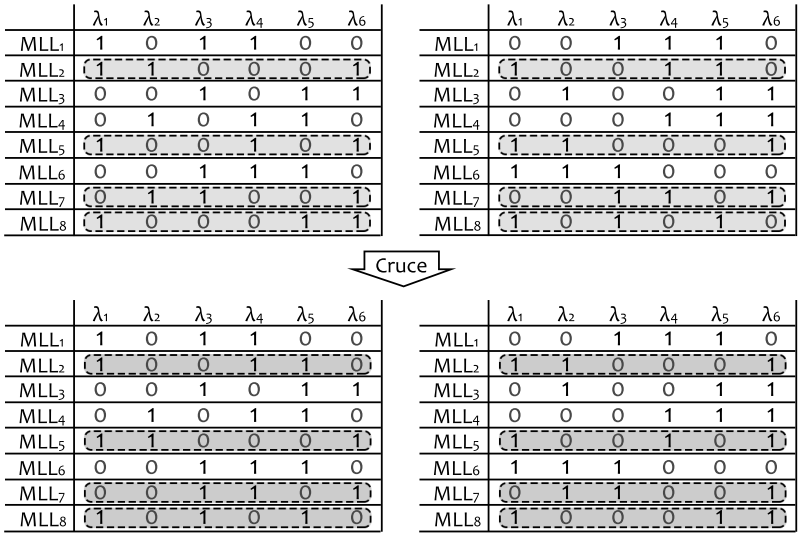

Figura 3. Operador de cruce uniforme

Operador de Mutación Cada individuo de la población se decide si muta aplicándole cierta probabilidad de mutación. El operador de mutación intercambia, para cada clasificador base, dos bits de valor distinto, haciendo que dicho clasificador base deje de tener en cuenta una etiqueta para pasar a considerar

otra. Este intercambio se realiza teniendo en cuenta la relación entre las etiquetas, buscando proyecciones de etiquetas relacionadas entre sí. Por tanto, el intercambio de bits se realiza como se observa en la Figura 4. Primero, se escoge un bit a 1 al azar dentro del clasificador base (a). Posteriormente, los pesos de mutación de los bits que están a 0 se inicializan a un valor $\varepsilon = 0{,}1$ (b), para que aun no estando relacionadas, todas las etiquetas tengan una pequeña probabilidad de mutar. Los pesos de cada uno de los bits a 0 se calculan acumulando los valores de correlación ϕ en valor absoluto entre dicha etiqueta y el resto de etiquetas activas (c). El peso de mutación de cada bit i que se encuentre a 0 será por tanto: $0{,}1 + \sum |\phi_{i,l}|$, denotando con l los bits activos. En base a estos pesos, se escoge uno de los bits a 0 para mutar (d) y se intercambia con el escogido inicialmente al azar (e). Así, se consigue que los bits que más relación tengan con el resto de bits a 1 tengan más probabilidad de activarse, favoreciendo conjuntos de etiquetas relacionadas. Los individuos generados son siempre factibles, porque el número de bits activos se mantiene.

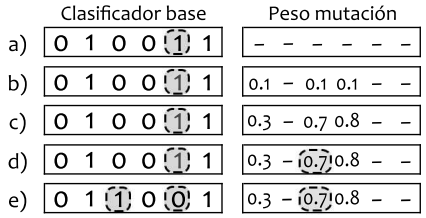

Figura 4. Operador de mutación basado en la relación entre etiquetas

3.4. Algoritmo Evolutivo

Para tener la seguridad de que el mejor individuo en la última generación es el mejor individuo visitado, el algoritmo evolutivo se basa en un algoritmo generacional con elitismo. Para ello, en cada generación se mantienen todos los hijos, excepto si el mejor padre es mejor que todos los nuevos individuos, que sustituirá al peor de estos. Para hacer más eficiente el proceso evolutivo, se almacenan el *fitness* de los individuos y todos los clasificadores base entrenados. Así, en los casos que se repitan individuos no habrá que volver a generar el *ensemble*, sino que se obtiene el *fitness* directamente. De igual manera, antes de entrenar un clasificador base, se obtiene si ya se entrenó con anterioridad. Finalmente, se obtiene el mejor individuo.

4. Experimentación

El propósito de la experimentación es comparar EME con otros algoritmos del estado del arte en clasificación multi-etiqueta. Los experimentos se han realizado

sobre 10 *datasets* de referencia de distintos dominios, como categorización de textos, multimedia, o biología. En la Tabla 1 se muestran características de los distintos conjuntos de datos utilizados, incluyendo el dominio del *dataset*, el número de instancias (p), número de etiquetas (q), número de atributos (d), cardinalidad (número medio de etiquetas por instancia), densidad (cardinalidad entre número de etiquetas) y número de combinaciones de etiquetas distintas que aparecen en el *dataset* (*distinct*). En la tabla, los *datasets* se encuentran ordenados por complejidad ($p \times q \times d$), según [10].

Tabla 1. Características de los *datasets* utilizados

Dataset	Dominio	p	q	d	Cardinalidad	Densidad	*Distinct*
CHD_49	Medicina	555	49	6	2.580	0.430	34
Water_quality	Otros	1060	16	14	5.072	0.362	825
Emotions	Audio	593	6	72	1.869	0.311	27
Birds	Audio	645	12	260	1.014	0.053	133
Yeast	Biología	2417	14	103	4.237	0.303	198
Scene	Imágenes	2407	6	294	1.074	0.179	15
Plant	Biología	948	12	440	1.080	0.089	32
Human	Biología	3108	14	440	1.190	0.084	85
Genbase	Biología	662	27	1186	1.252	0.046	32
Medical	Texto	978	45	1449	1.245	0.028	94

Los parámetros utilizados para EME son los siguientes: población de 50 individuos y máximo de 100 generaciones, probabilidades para realizar cruce y mutación de 0.8 y 0.2 respectivamente, cada *ensemble* está compuesto por $2q$ clasificadores base, que son LP con árbol de decisión J48, cada clasificador base tiene 3 etiquetas activas, y el umbral para la votación es 0.5. El resto de algoritmos ejecutados son todos los *ensembles* multi-etiqueta descritos en los antecedentes, y BR y LP como algoritmos clásicos de clasificación multi-etiqueta, todos con los parámetros por defecto según sus autores.

Para llevar a cabo los experimentos, se ha realizado una partición en *5-folds*, ejecutando los algoritmos con 10 semillas distintas, realizándose así 50 ejecuciones para cada algoritmo sobre cada *dataset*. Las métricas sobre las que se ha evaluado son: *Hamming loss* y *subset accuracy* como métricas basadas en ejemplos, y como métricas basadas en etiquetas *precision*, *recall*, *FMeasure*, *specificity* y *accuracy* en su aproximación macro. Se ha escogido el enfoque macro porque da el mismo peso a todas las etiquetas, y por tanto, estará más influenciada por categorías minoritarias en los casos que existan (que suelen ser las más interesantes).

5. Resultados y Discusión

Tras obtener los resultados se calculan, para cada métrica y *dataset*, el *ranking* de los algoritmos, donde el mejor obtiene un *ranking* de 1, el siguiente de 2,

y así sucesivamente. Después para cada métrica se calcula el *ranking* medio, promediando los valores de cada algoritmo en todos los *datasets* (Tabla 2).

Tabla 2. Ranking medio de los algoritmos para cada métrica

	EME	**ECC**	**EPS**	**HOMER**	**MLS**	**RA*k*EL**	**BR**	**LP**
Hamming loss	2,40	**1,80**	3,60	5,90	5,50	4,80	4,50	7,50
Subset accuracy	**2,60**	3,00	3,05	5,35	6,35	5,50	5,25	4,90
Precision$_{\text{macro}}$	**1,80**	2,40	4,60	5,90	5,35	4,80	4,35	6,80
Recall$_{\text{macro}}$	3,90	5,50	5,40	3,80	4,65	3,80	**3,35**	5,60
FMeasure$_{\text{macro}}$	**3,10**	4,90	5,40	3,90	4,75	4,10	3,75	6,10
Specificity$_{\text{macro}}$	2,85	**1,90**	3,10	5,90	5,30	5,05	5,20	6,70
Accuracy$_{\text{macro}}$	2,40	**1,80**	3,60	5,90	5,50	4,80	4,50	7,50
Valor promedio	**2,72**	3,04	4,11	5,24	5,34	4,69	4,41	6,44

Como se observa en la tabla, EME es el mejor en 3 de las 7 métricas, ECC en otras 3 y BR en una. EME obtiene el mejor *ranking* en una métrica tan estricta como *subset accuracy*, donde las predicciones deben ser completamente correctas. Igual, para $precision_{macro}$ es el algoritmo que mejores resultados proporciona. En el caso del $recall_{macro}$, es BR el algoritmo de control, y EME queda cuarto, por delante de ECC (que queda en sexto lugar). El *ranking* obtenido por EME en cuanto a $recall_{macro}$ es razonable teniendo en cuenta que *precision* y *recall* son métricas contrapuestas, y el hecho de obtener buenos resultados en una suele conllevar bajos resultados en la otra. Aun así, EME mejora en este sentido a ECC porque queda delante en *precision* y *FMeasure* quedando ECC en *FMeasure* el cuarto. Por último, en los casos de *Hamming loss*, $specificity_{macro}$ y $accuracy_{macro}$, en que ECC es el algoritmo de control, EME queda siempre inmediatamente después.

Para saber si hay diferencias significativas en el rendimiento, se realiza un test de Friedman [4], un test estadístico no paramétrico para comparar varios algoritmos sobre varias muestras que compara el *ranking* medio. La hipótesis nula es que todos los algoritmos tienen el mismo rendimiento, y la hipótesis alternativa que sí existen diferencias significativas. Para cada métrica el test da un valor de χ^2, y si es mayor que el valor de una distribución χ^2 con 9 (10 *datasets* $-$ 1) grados de libertad y $p = 0{,}05$, la hipótesis nula se rechaza al 95 % de confianza. Una vez realizados los tests, se puede afirmar al 95 % de confianza que existen diferencias significativas para 5 de las 7 métricas evaluadas. En cambio, para $recall_{macro}$ y $FMeasure_{macro}$ (donde EME es el algoritmo de control), no existen diferencias significativas (Tabla 3).

Para los casos en los que existen diferencias significativas, se ha de realizar un post-test. El test de Holm compara el algoritmo de control (mejor *ranking*) con el resto, comparando cada *p-value* p_i con $\alpha/(c-i)$, siendo c el número de *datasets*. Ordena todos los *p-value*, y comienza a comparar por el *p-value* más significativo. Si p_1 está por debajo de $\alpha/(c-1)$, se rechaza la hipótesis nula y se pasa a comparar p_2 con $\alpha/(c-2)$, y así sucesivamente. Si alguna hipótesis nula no se rechaza, las hipótesis restantes tampoco. Al aplicar el test se obtienen los

Tabla 3. Resultados del test de Friedman

	Algoritmo de control	Friedman	χ^2_9	Conclusión
Hamming loss	ECC	40,933		Rechaza hipótesis nula
Subset accuracy	EME	23,050		Rechaza hipótesis nula
Precision$_{\text{macro}}$	EME	32,992		Rechaza hipótesis nula
Recall$_{\text{macro}}$	BR	9,508	16,919	Acepta hipótesis nula
FMeasure$_{\text{macro}}$	EME	11,058		Acepta hipótesis nula
Specificity$_{\text{macro}}$	ECC	32,792		Rechaza hipótesis nula
Accuracy$_{\text{macro}}$	ECC	40,933		Rechaza hipótesis nula

resultados que se muestran en la Tabla 4, donde se indica con • los algoritmos que tienen diferencias significativas con el algoritmo de control al 95 % de confianza, mientras que "-" significa que el rendimiento es estadísticamente igual.

Tabla 4. Diferencias significativas resultado del test de Holm

	EME	ECC	EPS	HOMER	MLS	RA*k*EL	BR	LP
Hamming loss	-	-	-	•	•	•	•	•
Subset accuracy	-	-	-	-	•	•	-	-
Precision$_{\text{macro}}$	-	-	•	•	•	•	•	•
Specificity$_{\text{macro}}$	-	-	-	•	•	•	•	•
Accuracy$_{\text{macro}}$	-	-	-	•	•	•	•	•

Excepto para el EME y ECC, existen diferencias significativas con el algoritmo de control en, al menos, una métrica, con una confianza del 95 %. Por tanto, se puede afirmar que el rendimiento de ambos algoritmos es estadísticamente el mismo y superior al del resto. Sin embargo, EME parece más consistente que ECC, pues en valor promedio de *ranking*, EME tiene un valor de 2.72 mientras el de ECC es de 3.04. Es decir, aún teniendo ambos estadísticamente el mismo rendimiento, EME obtiene mejor valor de *ranking* en promedio de todas las métricas.

6. Conclusiones

En este artículo se ha propuesto el diseño de *ensembles* de clasificadores multi-etiqueta basados en proyecciones del espacio de etiquetas mediante el uso de algoritmos evolutivos. Los experimentos sobre un conjunto diverso de *datasets* y métricas, dan como resultado que existen diferencias significativas entre el algoritmo propuesto y otros algoritmos del estado del arte, siendo EME uno de los mejores, únicamente comparable a ECC, pero más consistente que este. Como líneas de trabajo futuro se pretende usar otros clasificadores base en lugar de LP con J48, tener un número distinto de etiquetas activas k en cada clasificador base, probar otros métodos de votación o tener en cuenta la diversidad en el *ensemble*.

7. Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2014-55252-P del Ministerio de Economía y Competitividad y fondos FEDER.

Referencias

1. Antenreiter, M., Ortner, R., Auer, P.: Combining classifiers for improved multilabel image classification. In: 1st Workshop on Learning from Multilabel Data (MLD) Held in Conjunction with ECML/PKDD. pp. 16–27 (2009)
2. Barutcuoglu, Z., Schapire, R., Troyanskaya, O.: Hierarchical multi-label prediction of gene function. Bioinformatics 22, 830–836 (2006)
3. Boutell, M., Luo, J., Shen, X., Brown, C.: Learning multi-label scene classification. Pattern recognition 37, 1757–1771 (2004)
4. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research (7), 1–30 (2006)
5. Gibaja, E., Ventura, S.: Multi-label learning: a review of the state of the art and ongoing research, WIREs Data Mining Knowl Discov 2014. doi: 10.1002/widm.1139
6. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 22–30 (2004)
7. Loza, E., Fürnkranz, J.: Efficient multilabel classification algorithms for large-scale problems in the legal domain. In: Semantic Processing of Legal Texts. vol. 6036, pp. 192–215 (2010)
8. Madjarov, G., Kocev, D., Gjorgjevikj, D., Deroski, S.: An extensive experimental comparison of methods for multi-label learning. Pattern Recognition 45(9), 3084–3104 (2012)
9. Read, J.: A pruned problem transformation method for multi-label classification. In: Proceedings of the NZ Computer Science Research Student Conference. pp. 143–150 (2008)
10. Read, J.: Scalable multi-label classification. PhD Thesis, University of Waikato (2010)
11. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Machine Learning 85(3), 335–359 (2011)
12. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learningl (37), 297–336 (1999)
13. Tang, L., Liu, H.: Scalable learning of collective behavior based on sparse social dimensions. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)
14. Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., Vlahavas, I.: Correlation-based pruning of stacked binary relevance models for multi-label learning. In: 1st International Workshop on Learning from Multi-Label Data (MLD'09). pp. 101–116 (2009)
15. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of the ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08) (2008)
16. Tsoumakas, G., Katakis, I., Vlahavas, I.: Data Mining and Knowledge Discovery Handbook, Part 6, chap. Mining Multi-label Data, pp. 667–685. Springer (2010)

17. Tsoumakas, G., Katakis, I., Vlahavas, I.: Random k-labelsets for multi-label classification. IEEE Transactions on Knowledge and Data Engineering 23(7), 1079–1089 (2011)
18. Yang, Y.: An evaluation of statistical approaches to text categorization. Information Retrieval (1), 69–90 (1999)
19. Zhu, S., Ji, X., Xu, W., Gong, Y.: Multi-labelled classification using maximum entropy method. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 274–281 (2005)