UCLV
Universidad Central
"Marta Abreu" de Las Villas

MFC
Facultad de Matemática
Física y Computación

Departamento de Ciencia de la Computación

TRABAJO DE DIPLOMA

Mejoras al método ML-kNN utilizando la Teoría de los Conjuntos Aproximados

Autor: Gabriela Pérez Hernández

Tutores: MSc. Marilyn Bello García

Dra. María Matilde García Lorenzo

Santa Clara, junio, 2018

Dedicatoria

A mis padres.

Agradecimientos

Agradezco en primer lugar a mis padres, por su amor, por ayudarme a convertirme en la mujer que soy hoy, por sacrificarme por mí, por ser duros cuando lo he necesitado, y por sobre todo por ser mis guías y paradigma a seguir en las metas que me trazo en la vida.

A mis abuelos, quienes me han mimado toda una vida, me han compartido sus consejos, y a quienes nunca voy a olvidar.

A mi esposo, quien ha estado ahí aguantando mis resabios durante ocho años, en el IPU y en la universidad, por quererme como soy, por compartir con mis amigos, aunque no sea su mundo, en fin, por ser mi apoyo.

A mi hermana Daniela, gracias por ser mi amiga, mi confidente, y por los buenos años que pasamos viviendo juntas.

A mis tíos Coby, Pepe y Bety, que son como unos segundos padres, por las noches de dominó para celebrar haber salido bien en una prueba, y por estar ahí siempre que los necesité.

A mis amigas de toda la vida Jennifer y Beatriz, por los años inolvidables en el IPU, que no importa que los caminos tomen rumbos diferentes siempre están ahí, por compartir juntas todos los momentos importantes en nuestras vidas.

A mi amiga Chabeli, que ha demostrado que la distancia no importa si se trata de amistad, gracias porque sé que no interesa el tiempo que pasemos sin vernos incluso sin hablar, siempre sé que puedo te puedo hablar sin tabúes y que me vas a apoyar, gracias por el cariño, y por los consejos de amiga.

A mi amiga Diacny, por demostrarme que la vida es corta y hay que disfrutar cada momento al máximo como si no hubiera un mañana, por las innumerables que me ha hecho reír por cualquier bobería para acabar con un mal momento, además por encima de eso te agradezco el estar ahí siempre que te he necesitado.

A Ana Laura y Leygui Mirtha, por brindarme su ayuda y estar presente en los momentos importantes.

A todas mis amigas gracias por sobre todo por hacerme creer que mis consejos son importantes en sus vidas, en fin, por ser mis amigas.

A todos mis compañeros de estudio en especial a Darío y Alejandro, que los he molestado un montón de veces y nunca he recibido un no por respuesta, también a Junior (alias la tía), por preparar los momentos inolvidables que son los viajes en grupo.

A mis tutoras Marilyn y María Matilde, por toda su dedicación y tiempo brindado, sin su ayuda no hubiera sido posible este sueño.

A todos los profesores que me impartieron clases durante estos cinco años y que me formaron como profesional, en especial, a Leticia y a Beatriz María.

En fin, a todas las personas que conozco que de una forma u otra han contribuido a mi formación como profesional y como ser humano.

Resumen

La clasificación multi-etiqueta es un campo de creciente actualidad dentro del aprendizaje automatizado, hace referencia al problema de relacionar un objeto con varias etiquetas a la vez. La misma ha sido tratada desde el enfoque de la transformación de problemas y la adaptación de algoritmos.

Multi-Label k-Nearest Neighbour (ML-kNN) es un algoritmo que a pesar de su sencillez ha reportado buenos resultados en la solución de problemas multi-etiquetas, sin embargo, no ha sido adaptado para tratar problemas con presencia de inconsistencia.

En este trabajo se propone una adaptación al algoritmo ML-kNN basada en la Teoría de los Conjuntos Aproximados Extendida para dar solución a esta problemática. La modificación fundamental está en la inclusión de un algoritmo para el cálculo de los pesos a partir de la metaheurística *Particle Swarm Optimization* (PSO) y la Medida de Calidad de la Similaridad. Esta nueva propuesta fue incorporada a la biblioteca Mulan.

En el estudio experimental realizado se demuestra que la nueva propuesta supera en eficacia al algoritmo ML-kNN, en particular en conjuntos de datos con presencia de inconsistencia.

Abstract

Multi-label classification refers to the problem of associating an object with multiple labels. This problem has been successfully addressed from the perspective of problem transformation and adaptation of algorithms.

Multi-Label k-Nearest Neighbor (ML-kNN) is a lazy learner that has reported excellent results, still there is room for improvements.

In this paper, we propose a modification to the MLkNN algorithm for the solution to problems of multi-label classification based on the Extended Rough Set Theory. More explicitly, the key modifications are focused in obtaining the relevance of the attributes when computing the distance between two instances, which are obtained using a heuristic search method and a target function based on the quality of the similarity.

Experimental results using synthetic dataset have shown promising prediction rates. It is worth mentioning the ability of our proposal to deal with inconsistent scenarios, a main shortcoming present in most state-of-the-art multi-label classification algorithms.

Tabla de contenidos

Introducción

En el mundo de hoy la información juega un papel de primer orden, podría catalogarse como el producto del siglo XXI, y quien la posea tendrá en sus manos el dominio del mundo.

Dicho de esta forma parece muy fácil apoderarse de la información dado que fluye en todos los sentidos y ámbitos, es generada por usuarios, instituciones, empresas, publicándose cada día miles de entradas en blogs, foros y redes sociales, las bases de datos están saturadas de información que crece cada día exponencialmente. Sin embargo, eso no es un problema dado que el desarrollo ha llevado a que se almacena todo sin tomar en cuenta qué es verdaderamente importante y qué no. El problema está en qué pasa si se quiere en un negocio satisfacer las necesidades de los clientes teniendo en cuenta preferencias particulares de estos; se necesita de mecanismos capaces no sólo de almacenar y recuperar eficientemente los datos, sino también procedimientos que permitan obtener información y conocimiento útil a partir de la enorme cantidad de datos que se tiene.

En este enfoque tiene su área de trabajo la Minería de Datos (Data Mining, *DM*), la cual se dedica al análisis de la información que ha sido almacenada en bases de datos, y se puede definir como un proceso de búsqueda y descubrimiento de patrones en datos. Este proceso debe ser automático y el objetivo fundamental es descubrir patrones que tengan algún significado útil (Witten, Frank and Hall, 2011). La idea es que estos patrones, o conceptos, ayuden en procesos de predicción como el antes ejemplificado.

La disciplina DM recoge una amplia gama de tareas, aunque la mayor parte de sus técnicas se enmarcan en el Aprendizaje Automático (Machine Learning, *ML*) (Witten, Frank and Hall, 2011), el cual se define como un subcampo de la Inteligencia Artificial (IA) que se ocupa de aquellos programas capaces de aprender a partir de la experiencia (Russell and Norvig, 1996). Dentro del ML encontramos el aprendizaje inductivo, cuyo objetivo es encontrar conceptos generales y descripciones. A partir de ejemplos, se adquiere conocimiento, se determina de alguna forma las similitudes entre estos ejemplos y se generalizan estas similitudes a todos los ejemplos del mismo concepto. Dicho de otra forma, la inducción consiste en formular afirmaciones que explican los hechos dados y estas pueden aplicarse a hechos que no hayan sido vistos. Estas afirmaciones pueden expresarse como patrones, y estos patrones se

representan por vectores, ecuaciones, árboles, reglas o enunciados lógicos. Muchos problemas de inducción se pueden describir como sigue: se parte de un conjunto de entrenamiento de ejemplos preclasificados, donde cada ejemplo (también llamado observación o caso) se describe por un vector de valores para rasgos o atributos, y el objetivo es formar una descripción que pueda ser usada para clasificar con alta precisión ejemplos no vistos previamente (Witten, Frank and Hall, 2011). Dentro del aprendizaje inductivo se distinguen tres paradigmas clásicos de aprendizaje: el aprendizaje supervisado, el no supervisado y el aprendizaje con refuerzo (Nilsson, 1996).

Las tareas de minería de datos usando técnicas de ML se agrupan en dos categorías: tareas descriptivas y tareas predictivas. Las primeras tratan de obtener información acerca de la estructura de los datos almacenados. Dentro de este grupo encontramos las tareas de agrupamiento y las tareas de asociación.

Por otro lado, las tareas predictivas parten de información ya almacenada y predicen un comportamiento. Dentro de estas tareas se encuentran la regresión y la clasificación.

La regresión es una tarea dentro del aprendizaje supervisado, en esta los ejemplos se presentan como un conjunto de pares de elementos, que representan las entradas y la salida esperada del clasificador. La salida es un valor real y el objetivo es aprender una función que represente la correspondencia existente en los ejemplos. En la clasificación los ejemplos se presentan de igual forma que en la regresión como un conjunto de pares de elementos que representan las entradas y la salida esperada, siendo la salida en este caso la clase a la que pertenece el ejemplo dado y el objetivo es determinar la clase de un nuevo problema a partir de un conjunto de aprendizaje conformado por ejemplos del área específica (Han and Kamber, 2006).

Dicho de otra forma, el problema de clasificación clásica consiste en asociar a cada patrón de entrenamiento una etiqueta de un conjunto de etiquetas existentes.

Sin embargo, en el caso de la clasificación multi-etiqueta (Multi-Label Classification, *MLC)* pueden existir patrones a los que se le asocie más de una etiqueta.

Esta problemática ha tenido un gran crecimiento en los últimos años en problemas como la clasificación de textos (Charte *et al.*, 2015) y multimedia (Briggs *et al.*, 2012), la predicción

de funciones de genes y proteínas (Chou, Wu and Xiao, 2011), el diagnóstico médico (Shao *et al.*, 2013), la predicción de interacciones entre medicamentos (Xiao *et al.*, 2015) y la categorización de sonidos (Cakir *et al.*, 2015).

Los problemas de MLC han sido abordados desde dos perspectivas fundamentales (Gibaja and Ventura, 2014): la *transformación de problemas* y la *adaptación de algoritmos*.

Los métodos de *transformación de problemas* transforman un problema multi-etiqueta en un problema de clasificación clásica, obteniendo de esta forma uno o varios conjuntos de datos de una sola etiqueta que puedan ser tratados con modelos de clasificación tradicionales.

Por otro lado, los métodos de *adaptación de algoritmos* adaptan técnicas de clasificación clásica al enfoque multi-etiqueta. La gran mayoría de las técnicas de clasificación clásicas existentes en la literatura han sido extendidas para trabajar directamente el enfoque multi-etiqueta. Los modelos representativos, así como algunas de sus propuestas son: redes neuronales (Nam *et al.*, 2014; Wang *et al.*, 2016), árboles de decisión (Bi and Kwok, 2011), kNN (Spyromitros, Tsoumakas and Vlahavas, 2008).

La esencia del aprendizaje basado en instancias es retornar como solución a un problema, la solución conocida a un problema similar. El método *k*-Vecinos Más Cercanos (k-Nearest Neighbor, kNN) constituye un algoritmo clásico de esta forma de solución a problemas y ha sido empleado en problemas de clasificación y regresión. El método básicamente consiste en comparar la nueva instancia a clasificar con los casos existentes del problema en cuestión, recuperando los *k* casos más cercanos, lo cual depende del parecido entre los atributos del nuevo caso con los casos de la muestra de aprendizaje o entrenamiento. Como resultado del mismo se devuelve la clase mayoritaria de aquellos *k* casos más cercanos a él.

En el enfoque multi-etiqueta el principal o clásico que respalda esta categoría de algoritmos es ML-kNN (Multi-Label k-Nearest Neighbor*)* (M. Zhang and Zhou, 2007), su prestigio se debe a su sencillez y a su rendimiento. Sin embargo, este algoritmo no está adaptado para resolver problemas con presencia de inconsistencia.

En (Filiberto and Bello, 2010; Filiberto, 2011; Coello *et al.*, 2017) se proponen adaptaciones al método *k*NN para tratar problemas con presencia de inconsistencia, para darle solución, se proponen la Medida de la Calidad de la Similaridad (Filiberto and Bello, 2010). Esta medida

representa el grado de similaridad entre los objetos de un sistema de decisión heterogéneo y constituye una herramienta importante para el análisis inteligente de los datos. También en (Coello *et al.*, 2017) proponen la medida de la Calidad de la Similaridad Borrosa, esta otra medida cuantifica a partir del uso de relaciones borrosas la fortaleza de la relación de semejanza entre dos objetos.

De lo anteriormente expuesto se derivó el siguiente **problema científico**:

Necesidad de nuevos métodos capaces de dar respuesta a la gran variedad de problemas de clasificación multi-etiqueta existentes, en particular aquellos problemas donde hay presencia de inconsistencia.

Para contribuir a la solución del problema científico antes plateado, se formuló la siguiente **hipótesis** general de la investigación:

La utilización de la Medida de la Calidad de la Similaridad en el algoritmo ML-kNN es factible para tratar problemas en presencia de inconsistencia.

En conformidad con la hipótesis de investigación identificada, el **objetivo general** de la investigación consiste en:

Adaptar el algoritmo MLkNN para tratar problemas con presencia de inconsistencia a partir de la Teoría de los Conjuntos Aproximados Extendida.

Este objetivo general fue desglosado en los siguientes **objetivos específicos**:

- Realizar un análisis crítico de los métodos de adaptación de algoritmos existentes para clasificación multi-etiqueta.
- Adaptar el método ML-kNN para tratar problemas con presencia de inconsistencia.
- Incorporar las modificaciones realizadas a ML-kNN en la biblioteca Mulan.
- Evaluar la nueva propuesta.

Las **tareas de investigación** trazadas son las siguientes:

- Revisión de los métodos de adaptación de algoritmos existentes en la literatura para la clasificación multi-etiqueta.

- Estudio de la biblioteca Mulan.
- Determinación de las modificaciones a desarrollar para resolver problemas con presencia de inconsistencia.
- Implementación de las modificaciones a desarrollar al algoritmo ML-kNN en la biblioteca Mulan.
- Evaluación de la nueva propuesta.

La **justificación de la investigación** es:

La Medida de la Calidad de la Similaridad ha sido utilizada con buenos resultados en la clasificación clásica para tratar problemas de inconsistencia. No siendo así, en la clasificación multi-etiqueta, de ahí que resulte de especial interés incursionar en este ámbito.

La **tesis está estructurada** en tres capítulos.

En el Capítulo 1 se abarca el problema de la clasificación multi-etiqueta, dándose una notación matemática y concentrándose en los métodos de adaptación de algoritmos, donde se tratarán las diferentes técnicas que han sido empleadas en este enfoque y ejemplos de cada una de ellas.

En el Capítulo 2 se propone una adaptación del método ML-kNN a partir de la Medida de Calidad de la Similaridad. Además, de su incorporación en la biblioteca de código abierto Mulan.

Finalmente, en el Capítulo 3 se hace una evaluación de la nueva propuesta a partir de las métricas utilizadas frecuentemente para medir el rendimiento de los algoritmos de clasificación multi-etiqueta.

Este documento culmina con las conclusiones, seguidamente las recomendaciones derivadas de la investigación y finalmente la bibliografía consultada.

Capítulo 1 Métodos de clasificación multi-etiqueta mediante adaptación de algoritmos

En este capítulo se realiza una breve reseña sobre la clasificación multi-etiqueta y su notación matemática. Además de un exhaustivo estudio de las técnicas existentes para resolver problemas multi-etiqueta desde el enfoque de los métodos de adaptación de algoritmos, las métricas usadas para medir el comportamiento de estos clasificadores, así como los principales campos de aplicación.

1.1 Clasificación multi-etiqueta

La clasificación se ha convertido en una de las principales áreas de interés dentro del aprendizaje automático. La misma tiene su fundamento en encontrar una función denominada "clasificador", que recibe como entrada los atributos de un determinado patrón y retorna como salida la clase que se asocia a dicho patrón.

Específicamente en la clasificación multi-etiqueta nos encontramos la disyuntiva en la que los patrones de entrada no están asociados a un único patrón de salida, sino que se encuentran relacionados con varios. Esto se pone de manifiesto en numerosos problemas de actualidad (Chou, Wu and Xiao, 2011; Briggs *et al.*, 2012; Shao *et al.*, 2013; Cakir *et al.*, 2015; Charte *et al.*, 2015; Xiao *et al.*, 2015)

Estos problemas tienen en común que un objeto (patrón o ejemplo) puede pertenecer a más de una clase (etiqueta) simultáneamente, no satisfaciéndose la restricción del aprendizaje clásico de solamente una etiqueta por patrón. Para representar este hecho, las etiquetas son variables binarias que indican la pertenencia a cada una de las clases de interés de forma análoga al aprendizaje multiclase, pero con la diferencia de que un ejemplo puede tener más de un valor binario activo.

De esta forma se puede sintetizar que el objetivo en la clasificación multi-etiqueta es hallar modelos capaces de asignar a un nuevo objeto de un dominio determinado las etiquetas que mejor lo describan.

1.1.1 Notación

Dado un conjunto no vacío de etiquetas $L = \{l_1, l_2, \dots, l_m\}$, donde m es el número máximo de etiquetas que puede tener asociadas un patrón. Sea $D = \{D_i : i = 1..n\}$ conjunto de datos

multi-etiqueta, donde cada elemento estará compuesto por un par (X_i, Y_i); siendo X_i el vector de características asociado al patrón e $Y_i \subseteq L$ el subconjunto de etiquetas asociadas con el patrón i. De esta forma $|Y_i|$ será el número de etiquetas asociadas al patrón i.

En la clasificación multi-etiqueta las etiquetas son binarias, o sea, los conjuntos de etiquetas se pueden presentar de dos formas diferentes. Se puede tener Y_i como un vector binario, donde sus elementos sean ceros o unos; o también como una lista de etiquetas.

Dadas estas características la tarea de aprendizaje automático consistirá en encontrar una función o clasificador multi-etiqueta $h : X \rightarrow P \subseteq L$ que para cada patrón i realice una bipartición del conjunto de etiquetas L que minimice la diferencia entre el conjunto de etiquetas predicho, P, y el conjunto de etiquetas que realmente tiene asociado el patrón (Ávila Jiménez, 2013).

1.2 Métodos de adaptación de algoritmos

Entre los métodos de adaptación de algoritmos existentes en la literatura se pueden encontrar: máquinas de vectores soporte (Wan and Xu, 2007; Li and Xu, 2009; Xu, 2013), árboles de decisión (Bi and Kwok, 2011), redes neuronales (Nam *et al.*, 2014; Wang *et al.*, 2016), kNN (Spyromitros, Tsoumakas and Vlahavas, 2008), clasificación asociativa (Thabtah, Cowling and Peng, 2004, 2006; Rak, Kurgan and Reformat, 2008), métodos probabilísticos (Ghamrawi and McCallum, 2005; Wang, Huang and Zhu, 2008; Bielza, Li and Larrañaga, 2011), así como algoritmos bioinspirados (Chan and Freitas, 2006; Vallim *et al.*, 2008; Khan and Rauf, 2017).

1.2.1 Máquinas de vectores soporte

Las máquinas de vectores soporte (*Support Vector Machines*, SVM) se han usado en problemas de clasificación binaria y multi-etiqueta, aunque no son fácilmente adaptables.

El algoritmo *Rank-SVM*, abordado en (Elisseeff and Weston, 2001), es una adaptación del algoritmo SVM para trabajar con problemas multi-etiqueta de forma directa, obteniendo buenos resultados en el etiquetado de escenas naturales (Boutell *et al.*, 2004).

En (Xu, 2013) y (Wang *et al.*, 2014) se exponen dos adaptaciones de este algoritmo conocidas como *Rank-CVM* y *SCRank-SVM* respectivamente. En ambas se trata de mejorar tanto el rendimiento en la clasificación como de reducir su complejidad computacional.

Otro ejemplar de las SVM adaptado al enfoque multi-etiqueta es *Triple Class SVM*, propuesto por Wan y Xu en (Wan and Xu, 2007). Este algoritmo aplica la técnica de uno contra todos y a los patrones con dos etiquetas los considera como pertenecientes a una clase intermedia entre las dos. Estos mismos autores proponen en (Li and Xu, 2009) el algoritmo *Double Label SVM*, similar al anterior, pero optimizado para que su ejecución sea más rápida.

1.2.2 Árboles de decisión

Los árboles de decisión (*Decision Trees*, DT) son una de las estructuras más sencillas usadas para la clasificación, por esto se han utilizado frecuentemente en problemas multi-etiqueta.

En (Clare and King, 2001) se propone el algoritmo ML-C4.5 basado en el algoritmo C4.5 clásico, pero modificado con el fin de poder trabajar con varias etiquetas.

Los *Alternate Decision Tree* (ADT) son tratados en (Freud and Mason, 1999), siendo una generalización de los árboles de decisión tradicionales como una alternativa al uso de técnicas de *boosting* para mejorar la precisión de los clasificadores basados en árboles. Utilizando esta teoría de los ADT y el modelo AdaBoost.MH (Schapire and Singer, 1999) se propone ADTBoost.MH (Comité, Gilleron and Tommasi, 2003) para el caso multi-etiqueta.

ML-TREE (Wu *et al.*, 2014) es un algoritmo que induce un árbol con una estructura jerárquica, usando en cada nodo el enfoque Objeto Virtual de Aprendizaje (OVA) (Botero and Palomeque, 2014) y clasificadores de tipo SVM para ir dividiendo el conjunto de datos a medida que se generan nuevos hijos, en un proceso recursivo. La frecuencia con que dos etiquetas aparecen conjuntamente en las hojas que van generándose se usa como un indicador de la relevancia de la relación entre ellas, definiendo así un modelo automático de descubrimiento de las dependencias entre etiquetas.

Por otro lado, en (Dimitrovski *et al.*, 2010) se propone una combinación basada en árboles de decisión, que utiliza el paradigma *Predictive Clustering Trees* (PCT) (Cherman, Metz and Monard, 2010), en el que se considera que un árbol de decisión realiza una tarea de agrupamiento sobre los patrones de tal manera que todos los patrones en el mismo grupo tienen propiedades similares.

1.2.3 Redes neuronales

Las redes neuronales artificiales (*Artificial Neural Networks*, RNA) en general, y las redes de funciones de base radial (*Radial Basis Function Networks*, RBFNs) en particular, han demostrado su efectividad en problemas de clasificación, regresión y predicción de series temporales (Ojeda, 2015).

Esta técnica se encuentra entre las más utilizadas en el campo del aprendizaje automático, por ello existen un gran número de adaptaciones de redes neuronales para ser utilizadas con problemas multi-etiqueta directamente (Ojeda, 2015).

En (Crammer and Singer, 2003) se propone el algoritmo *Multi-label Multi-class Perceptron* (MMP), siendo una generalización del Perceptron Multicapa que permite trabajar con datos multi-etiqueta, y en el que la salida de la red neuronal es un ranking de etiquetas.

En (Zhang, 2006) se propone el algoritmo *Back Propagation Multi-Label Layer* (BP-MLL); este algoritmo parte del modelo *Perceptron* y el algoritmo de aprendizaje *Back-Propagation*. El aspecto clave de BP-MLL es la introducción de una función para el cálculo del error cometido adaptada al hecho de que cada muestra tiene asociadas varias etiquetas. Para encontrar el conjunto de etiquetas predichas se usa un parámetro como umbral de corte.

En (Grodzicki, Mańdziuk and Wang, 2008) se resuelve la dificultad del ajuste del umbral, incorporándole a la función de cálculo del error la forma que el umbral va adaptándose automáticamente durante el proceso de aprendizaje.

En (Zhang, 2009) se adaptan las RBFNs al enfoque multi-etiqueta resultando en el algoritmo ML-RBF. Este algoritmo utiliza una función de base radial como función de transferencia, básicamente la adaptación consiste en añadir una nueva capa a la red de tal forma que se encargue de separar la información que corresponde a cada etiqueta.

Extensiones de este algoritmo se proponen en (Agrawal *et al.*, 2014) con el objetivo de mejorar su rendimiento.

El algoritmo CCA-ELM se introduce en (Kongsorot and Horata, 2014). El algoritmo propone una metodología para adaptar una *Extreme Learning* Machine (ELM), un tipo de red neuronal con una sola capa oculta que se caracteriza por su rapidez en aprendizaje, a fin de que pueda procesar datos multi-etiqueta y para ello se recurre a un *Canonical Correlation Analysis*

(CCA) para determinar correlaciones entre los atributos de entrada y las etiquetas, generando un nuevo espacio en el que se combinan las entradas y las etiquetas.

En los trabajos (Oliveira *et al.*, 2008; Ciarelli and Oliveira, 2009) se proponen adaptaciones a las redes neuronales probabilísticas, donde la red devuelve la probabilidad de pertenencia a una etiqueta.

1.2.4 Clasificación asociativa

En la tarea de asociación las clases no están predefinidas de antemano, sino que se busca agrupar patrones que presentan características similares (Li, Han and Pei, 2001).

Entre los trabajos que se pueden destacar está el algoritmo *Multi-class Multi-label Associative Classification* (MMAC) desarrollado por (Thabtah, Cowling and Peng, 2004, 2006). El algoritmo va aprendiendo reglas sucesivamente mediante técnicas asociativas, y en cada iteración elimina del conjunto de entrada los patrones cubiertos por dichas reglas.

En (Thabtah, Cowling and Peng, 2005; Thabtah and Cowling, 2007) los mismos autores proponen el algoritmo *Ranked Multi-label Rule* (RMR). Este algoritmo funciona de manera similar al anterior, pero incluye una etapa de poda para eliminar patrones de entrenamiento redundantes compartidos por las reglas.

1.2.5 Métodos probabilísticos

Los métodos probabilísticos básicamente lo que hacen es aplicar técnicas estadísticas a los problemas multi-etiqueta.

Para tratar este tipo de problemas se han propuesto tres enfoques, el enfoque *Single Label* en el que cada término contribuye a la probabilidad de una sola etiqueta, el *Collective Multi-Label* (CML) en el que se tienen en cuenta las relaciones entre etiquetas y el *Collective Multi-Label with Features classifier* (CMLF) en el cual, además de considerarse las relaciones entre etiquetas, cada término puede contribuir a la probabilidad de más de una etiqueta.

En (Bielza, Li and Larrañaga, 2011; Akbarnejad and Baghshah, 2017) se propone un algoritmo basado en clasificadores bayesianos y en (Cheng, Hüllermeier and Dembczynski, 2010; Sucar *et al.*, 2014) se propone una extensión de los *Classifier Chains*. Ambas propuestas modelan las dependencias entre etiquetas calculando la distribución conjunta completa de todas ellas, determinando cuál es el encadenamiento óptimo.

1.2.6 Algoritmos bioinspirados

Los algoritmos bioinspirados se basan en emplear analogías con sistemas naturales o sociales para la resolución de problemas. Simulan el comportamiento de sistemas naturales para el diseño de métodos heurísticos no determinísticos de búsqueda, aprendizaje, comportamiento, etc. Este enfoque ha sido poco utilizado en clasificación multi-etiqueta.

En (Chan and Freitas, 2006) se crea un algoritmo llamado *Multi-Label Ant Miner* (MULAM). Este algoritmo utiliza el enfoque bioinspirado de Colonias de Hormigas (Dorigo, Caro and Gambardella, 1999), apoyándose además en el método *Ant-Miner* (Parpinelli, Lopes and Freitas, 2002) que permite la extracción de reglas. El proceso de extracción de reglas ocurre de forma tal que cada hormiga descubre un conjunto de reglas candidatas para resolver el problema planteado, encontrando en cada iteración del algoritmo, como mínimo, una sola regla y, como máximo, una regla por clase. De esta forma se eliminan del conjunto de datos los patrones cubiertos por las reglas que ya se han encontrado.

1.2.7 Algoritmos basados en los *k*-Vecinos Más Cercanos

Existen varias adaptaciones de esta técnica para solucionar problemas de clasificación multi-etiqueta, entre ellas el algoritmo *Multi-Label kNN* (ML-kNN**)** descrito en (M. Zhang and Zhou, 2007).La idea básica del mismo consiste en seleccionar los *k* vecinos más cercanos para cada instancia a clasificar y seguidamente generar un ranking de las etiquetas que tienen asignadas y usar dicha información para calcular la probabilidad de aparición de cada etiqueta en la muestra a clasificar. Para determinar si una instancia pertenece a una clase o no, se calculan dos probabilidades. Primeramente, una probabilidad a priori, buscando la frecuencia de cada etiqueta en toda la población de entrenamiento, y luego, una probabilidad a posteriori, hallada con las etiquetas de la vecindad de la muestra a clasificar. Este algoritmo utiliza la función *Heterogeneous Euclidean-Overlap Metric* (HEOM) (AHA, D.K. and ALBERT, 1991), por lo que no hace uso de la información adicional proporcionada por los valores de los atributos nominales.

En (Cheng and Eyke Hullermeier, 2009) se proponen dos algoritmos basados en ML-kNN. Estos algoritmos tienen en cuenta la correlación entre las etiquetas, considerando las etiquetas asociadas a los *k* vecinos más cercanos de la instancia a clasificar como características adicionales de dicha instancia. También en (Younes, Abdallah and DenÃžeux, 2008) se

propone una generalización del ML-kNN donde la dependencia entre las etiquetas es considerada. Este usa un principio de máximo a posteriori global ya que considera la frecuencia de todas las etiquetas en los *k* vecinos más cercanos de todas las instancias de entrenamiento, en lugar de considerar solamente la cantidad de vecinos de la etiqueta a ser asignada.

Otro modelo es *Multi-label text categorization based on fuzzy similarity and k-nearest neighbors* (FSKNN) es un modelo propuesto por (Jiang, Tsai and Lee, 2012) para la clasificación de textos. Este algoritmo utiliza una medida de similitud borrosa para calcular la similitud entre un documento y cada categoría (etiquetas) de los documentos de entrenamiento y luego agrupar el conjunto de instancias de entrenamiento en *clusters*. Uno de los problemas asociados con el enfoque *k*NN es el costo computacional que demanda la búsqueda de los *k* vecinos más cercanos para todas las instancias de entrenamiento por lo que se emplea la similitud borrosa para hacer una preselección en la comparación.

El rendimiento de estos métodos se ve afectado principalmente por la distancia a emplear en la búsqueda de los *k* vecinos más cercanos.

Durante el entrenamiento pueden producirse errores en el etiquetado o pueden aparecer patrones ruidosos por problemas en la recopilación de los datos. Estos prototipos suelen aparecer en zonas cercanas a las regiones de decisión e influyen negativamente en el entrenamiento ya que pueden disminuir la precisión de la clasificación supervisada (Cárdenas, Caballero and Bello, 2007).

1.3 Métricas existentes en la literatura para evaluar el rendimiento de los métodos en la clasificación multi-etiqueta

Se han propuesto diversas formas de determinar el rendimiento de un clasificador multi-etiqueta. Básicamente las métricas que existen se agrupan en métricas basadas en etiquetas y métricas basadas en ejemplos.

1.3.1 Basadas en etiquetas

Las métricas basadas en etiquetas (Tsoumakas and Vlahavas, 2007) evalúan la medida en cada etiqueta por separado, luego son promediadas para obtener un valor único según dos enfoques distintos, a los que se conoce como macro y micro.

Es necesario hacer alusión a los valores que forman la matriz de confusión, *Tp* es el número de patrones positivos clasificados correctamente, *Tn* el número de patrones negativos correctamente asignados, *Fp* aquellos patrones que son llamados falsos positivos y *Fn* falsos negativos.

El enfoque macro (1.1) primero calcula la métrica para cada etiqueta, y se hace la media por etiquetas para obtener el valor final, otorgando el mismo peso a todas las etiquetas.

$$B_{macro} = \frac{1}{l}\sum_{i=1}^{l} B(Tp_i, Tn_i, Fp_i, Fn_i) \quad (1.1)$$

En el enfoque micro (1.2) se agregan los valores de la matriz de confusión para las etiquetas, y luego se calcula la métrica con estos valores, se calcula una sola vez la métrica, otorgando, por tanto, una visión más general del rendimiento al no ponderarse por etiquetas, sino efectuando la evaluación sobre el volumen total de predicciones hechas por el clasificador.

$$B_{micro} = B(\sum_{i}^{l} Tp_i, \sum_{i}^{l} Tn_i, \sum_{i}^{l} Fp_i, \sum_{i}^{l} Fn_i) \quad (1.2)$$

La función B es una métrica binaria, las más usadas en la literatura son *Precision* y *Recall*. Para un clasificador, *Precision* (1.3) es la división de etiquetas correctamente asignadas entre todas las etiquetas asignadas por el clasificador, y *Recall* (1.4) es la división de las etiquetas que han sido bien clasificadas entre todas las etiquetas que son verdaderamente positivas.

$$Precision = \frac{Tp}{Tp + Fp} \quad (1.3)$$

$$Recall = \frac{Tp}{Tp + Fn} \quad (1.4)$$

Otra métrica también utilizada en el enfoque micro y macro es *Accuracy* (1.5) que se define como la proporción de etiquetas correctamente clasificadas con respecto al total de etiquetas. Es considerada una de las medidas más intuitivas para evaluar el rendimiento de un clasificador multi-etiqueta (Ojeda, 2015).

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \quad (1.5)$$

1.3.2 Basadas en ejemplos

Las métricas basadas en ejemplos tienen como entrada la clasificación de cada instancia, luego se promedia el valor de la métrica para obtener el valor medio de esta sobre todo el conjunto de entrenamiento.

Este conjunto de métricas se encuentra dividido en las llamadas métricas binarias, que pueden ser aplicadas a cualquier algoritmo de clasificación multi-etiqueta, y las métricas de ranking que son específicamente para algoritmos de ranking.

Dentro de las métricas binarias se encuentra *Subset Accuracy* (1.6), que no ha sido muy utilizada en la clasificación multi-etiqueta, dado que solo contabiliza como correctos el caso en que el conjunto real (Y_i) y el predicho (P_i) son exactamente iguales. Calcula el porciento de ejemplos que el clasificador ha acertado en todas sus etiquetas asociadas.

$$\text{Subset Accuracy} = \frac{1}{n}\sum_{i=1}^{n} \delta(P_i = Y_i) \quad (1.6)$$

$$\delta(x) = \begin{cases} 1 & \text{si x es verdadero} \\ 0 & \text{si x es falso} \end{cases}$$

Una de las medidas más utilizadas es *Hamming Loss* (1.7), esta medida obtiene la diferencia entre el conjunto de etiquetas reales y el conjunto de etiquetas predichas, por ello cuanto más se acerque su valor a cero mejor es el resultado que está aportando el clasificador. Sus buenos resultados están sustentados en el hecho de que tiene en cuenta cuando una etiqueta incorrecta es predicha, así como, cuando se omite una etiqueta del conjunto de salida del clasificador. Siendo $|P_i \Delta Y_i|$ la diferencia simétrica entre los dos conjuntos de etiquetas predichas y reales, este valor es dividido entre el número total de etiquetas, y luego se promedia.

$$\text{Hamming Loss} = \frac{1}{n}\sum_{i=1}^{n} \frac{|P_i \Delta Y_i|}{l} \quad (1.7)$$

Las métricas descritas en el epígrafe anterior *Precision, Accuracy y Recall* han sido extendidas para que tengan en cuenta la respuesta completa del clasificador. La métrica *Accuracy* (1.8) mide como la proporción de etiquetas predichas correctamente con respecto al total de etiquetas, y luego promedia el resultado de todas las instancias, *Precision* (1.9) es

el promedio de etiquetas acertadas frente a las predichas, y R*ecall* (1.10) es el promedio de etiquetas acertadas frente a las reales.

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap P_i|}{|Y_i \cup P_i|} \quad (1.8)$$

$$\text{Precision} = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap P_i|}{|P_i|} \quad (1.9)$$

$$\text{Recall} = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap P_i|}{|Y_i|} \quad (1.10)$$

En cuanto a las métricas de ranking se encuentra *Coverage* (1.11) que evalúa dada una instancia hasta qué punto hay que avanzar en el ranking *r* devuelto por el clasificador para esa instancia (r_i) hasta lograr cubrir todas las etiquetas asociadas a un ejemplo, promediando este valor para todas las instancias.

$$\text{Coverage} = \frac{1}{n}\sum_{i=1}^{n}(\max_{\lambda \in Y_i} r_i(\lambda) - 1) \quad (1.11)$$

La métrica *Ranking Loss* (1.12), es una medida en la que se cuenta como errores los casos en que para cada una de las parejas posibles de etiquetas relevantes e irrelevantes una etiqueta irrelevante (λ_a) aparece en el ranking primero que las etiquetas relevantes (λ_b), como en las anteriores medidas este valor se promedia para todas las instancias dadas.

$$\text{Ranking Loss} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i||\overline{Y}_i|}\left|(\lambda_a, \lambda_b): r_i(\lambda_a) > r_i(\lambda_b), (\lambda_a, \lambda_b) \in Y_i \times \overline{Y}_i\right| \quad (1.12)$$

Otra de las métricas más utilizadas es *Average Precision* (1.13), que nos indica el promedio de etiquetas con un ranking superior a una etiqueta $\lambda \in Y_i$ (Ecuación 1.13), es decir, nos indica cuanto hay que avanzar en promedio en el ranking hasta encontrar una etiqueta determinada.

$$\text{Average Precision} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i|}\sum_{\lambda \in Y_i}\frac{|\{\lambda' \in Y_i: r_i(\lambda') \leq r_i(\lambda)\}|}{r_i(\lambda)} \quad (1.13)$$

Donde *rank* es una función que para una instancia x y la etiqueta relevante y, perteneciente a Y_i, cuya posición es conocida, facilita el valor que en el ranking Z_i ocupa esa posición.

1.4 Aplicaciones de la clasificación multi-etiqueta

La clasificación multi-etiqueta se ha aplicado para resolver múltiples problemas reales, en múltiples dominios, como categorización de documentos, bioinformática, clasificación de imágenes, o medicina entre otras. A continuación, se abordan las principales particularidades de cada una de estas aplicaciones.

Clasificación de textos: Al clasificar un texto por categorías semánticas, por ejemplo, su temática, es bastante común que a un mismo documento le correspondan dos o más categorías. Por ejemplo, no es difícil encontrar un artículo científico en el área de Ciencias de la Computación que pueda ser categorizado como de *machine learning* y bases de datos simultáneamente. Los primeros trabajos se centraron en manejar esta mediante técnicas de clasificación clásicas. Problemas de clasificación de textos como la detección de spam, el análisis y clasificación automática de correo electrónico, detectar estados afectivos, identificar opiniones, etc, son abordados mediante clasificadores multi-etiqueta. Dado que fue una de las primeras aplicaciones de la clasificación multi-etiqueta existen varios corpus documentales que se suelen utilizar frecuentemente para evaluar nuevas propuestas, entre ellos se encuentran: el corpus textual de la agencia Reuters (RCV1) (Lewis *et al.*, 2004), el corpus de la ACM (Veloso, Goncalves and Zaki, 2007), etc.

Clasificación de imágenes: La idea básica es asociar automáticamente cada imagen a las clases o etiquetas a la que pertenece. Existen trabajos que tratan el reconocimiento de objetos se ha llevado un poco más allá, utilizándolo para reconocer rasgos faciales (Zong and Huang, 2011),

Bioinformática: El campo de la bioinformática es uno de los que más se ha beneficiado con la utilización de técnicas multi-etiqueta, ya que estas se han aplicado satisfactoriamente a problemas como la clasificación de proteínas o de genes, la determinación de la estructura de las proteínas o la multi-localización de proteínas. Existe el conjunto de datos genbase (Diplaris *et al.*, 2005), que se suele utilizar para comparar propuestas sobre este dominio de aplicación.

Medicina: En este campo el ejemplo más ilustrativo es el diagnóstico clínico ya que existen distintas enfermedades que comparten síntomas entre sí. Otro problema que se ha modelado mediante técnicas de aprendizaje multi-etiqueta es el de las interacciones entre medicamentos, siendo las diversas etiquetas las sustancias que presentan interacciones. También se ha aplicado a problemas como la acción de medicamentos anti-cancerígenos sobre distintos tipos de células tumorales (Zafra and Ventura, 2010).

Clasificación de música o sonidos: El problema de la categorización de sonidos consiste en clasificar sonidos, sobre todo musicales, según un criterio determinado. De la misma manera que ocurría con las imágenes, con mucha frecuencia es necesario modelar este problema con aprendizaje multi-etiqueta, porque las categorías en las que se agrupan las piezas musicales suelen solaparse entre sí. Se puede clasificar la música en géneros musicales, habiendo multitud de composiciones que se pueden considerar asociadas a varios géneros simultáneamente, también teniendo en cuenta los instrumentos que intervienen.

1.5 Conclusiones parciales del capítulo

Los métodos de clasificación multi-etiqueta se han utilizado en diferentes campos de aplicación como: la clasificación de textos y de imágenes, la bioinformática, la medicina y la clasificación de sonidos.

Entre las ventajas que presentan los métodos de adaptación de algoritmos podemos mencionar que son menos complejos computacionalmente y que la mayoría de las técnicas de clasificación clásicas han sido extendidas para trabajar directamente el enfoque multi-etiqueta.

De la revisión bibliográfica realizada se pudo comprobar la no existencia de métodos de adaptación de algoritmos para tratar la inconsistencia de los datos de una manera eficaz.

Capítulo 2 Adaptaciones del algoritmo ML-kNN para tratar problemas con presencia de inconsistencia

En este capítulo primeramente se hace una descripción del algoritmo ML-kNN, para luego entender las modificaciones hechas al mismo. Seguidamente se explican las dos propuestas que se plantea esta tesis.

2.1 Algoritmo ML-kNN

El modelo ML-kNN es un algoritmo perezoso derivado del tradicional algoritmo *k*NN, este es uno de los más utilizados en clasificación clásica. Este algoritmo determina si una instancia pertenece a una clase determinada por las k instancias más cercanas del conjunto de entrenamiento y examinando la clase a la que pertenecen estas.

Veremos algunas notaciones para entrar en detalles con el algoritmo: dada una instancia x y sus etiquetas asociadas $Y \subseteq \Upsilon$, tenemos que $\overrightarrow{y_x}$ es el vector categoría para x donde la *l*-ésima componente de $\overrightarrow{y_x}(l)(l \in \Upsilon)$ toma el valor 1 si $l \in \Upsilon$ y 0 en otro caso. Denotaremos $N(x)$ el conjunto de los *k* vecinos de x identificados en el conjunto de entrenamiento. Así, basados en el conjunto de etiquetas de estos vecinos el vector de conteo de membresía puede definirse como:

$$\overrightarrow{C_x}(l) = \sum \overrightarrow{y_a}_{a \in N(x)}(l), l \in \Upsilon \quad (2.1)$$

Donde $\overrightarrow{C_x}(l)$ cuenta el número de vecinos de x que pertenecen a la *l*-ésima clase. Para cada instancia t, ML-kNN primero identifica sus k vecinos más cercanos $N(t)$ en el conjunto de entrenamiento. Así H_1^l es el evento donde t tiene la etiqueta l, mientras H_0^l es el evento donde t que no tiene la etiqueta l. Además, el $E_j^l(l \in \Upsilon)$ denota el evento que entre los k vecinos hay exactamente j instancias con la etiqueta l. Sin embargo, basado en el vector de conteo de membresía $\overrightarrow{C_x}(l)$ el vector categoría $\overrightarrow{y_x}$ es determinado usando el siguiente principio "*maximum a posteriori*":

$$\overrightarrow{y_x}(l) = \text{argmax}_{b \in \{0,1\}} P\left(H_b^l \middle| E^l_{\overrightarrow{C_t}(l)}\right), l \in \Upsilon \quad (2.2)$$

Usando las reglas Bayesianas, la ecuación anterior puede rescribirse como:

$$\overrightarrow{y_x}(l) = \text{argmax}_{b\in\{0,1\}} \frac{P(H_b^l)P\left(E^l_{\overrightarrow{C_t}(l)} \middle| H_b^l\right)}{P(E^l_{\overrightarrow{C_t}(l)})}$$

$$= \text{argmax}_{b\in\{0,1\}} P(H_b^l)P\left(E^l_{\overrightarrow{C_t}(l)} \middle| H_b^l\right) \quad (2.3)$$

Anteriormente se dijo que H_1^l es el evento donde t tiene la etiqueta l y H_0^l es el evento donde t que no tiene la etiqueta l, de forma general H_b^l se calcula para cada etiqueta y es conocida como probabilidad *a priori*. Por tanto, $P(H_1^l)$ (ver Ecuación 2.4), es la probabilidad de que esté esa etiqueta, sin embargo, $P(H_0^l)$ (ver Ecuación 2.5), es la probabilidad de que no esté esa etiqueta.

$$P(H_1^l) = \frac{s + \sum_{i=1}^{n} \overrightarrow{y_{x_i}}(l)}{s \times 2 + n} \quad (2.4)$$

$$P(H_0^l) = 1 - P(H_1^l) \quad (2.5)$$

Donde,

s: parámetro de suavizado,

n: cantidad de instancias,

$\overrightarrow{y_{x_i}}$: vector de etiquetas para la instancia i.

También, como se observa en la Ecuación 2.2, además de $P(H_b^l)$, la probabilidad a posteriori definida anteriormente, se necesita otra probabilidad, $P\left(E^l_{\overrightarrow{C_t}(l)} \middle| H_b^l\right)$ (ver Ecuaciones 2.6 y 2.7), conocida esta como probabilidad *a posteriori*. Esta fórmula es calculada para cada uno de los *k* vecinos más cercanos y luego para cada una de las etiquetas por lo que $\overrightarrow{C_t}(l)$ en cada iteración se reemplaza por *j*, sería la etiqueta *j*-ésima.

$$P(E_j^l|H_0^l) = (s + c'[j])/(s \times (k + 1) + \sum_{p=0}^{k} c'[p] \quad (2.6)$$

$$P(E_j^l|H_1^l) = (s + c[j])/(s \times (k + 1) + \sum_{p=0}^{k} c[p] \quad (2.7)$$

Donde,

$c'[j]$: cantidad de los k vecinos más cercanos que no tienen la etiqueta j,

$c[j]$: cantidad de los k vecinos más cercanos que tienen la etiqueta j,

$c'[p]$: cantidad de etiquetas que no poseen los k vecinos más cercanos,

$c[p]$: cantidad de etiquetas que poseen los k vecinos más cercanos.

En general, para cada instancia no vista, se determina del conjunto de instancias de entrenamientos sus k vecinos más cercanos y después basado en la información estadística obtenida del conjunto de etiquetas de dichos vecinos, es decir, el número de instancia que pertenecen a cada clase, se utiliza el principio de "*maximum a posteriori*" para determinar el conjunto de etiquetas para la instancia no vista.

Este método de clasificación es conocido y estudiado en gran medida en la comunidad de reconocimiento de patrones, tanto por su sencillez como por su rendimiento. La definición de la función de distancia es fundamental para obtener una buena precisión en un conjunto de datos dado y diferentes funciones de distancia se han propuesto para aumentar el rendimiento.

2.2 Descripción de la primera propuesta

Para poder tratar problemas con presencia de inconsistencia se propone un primer algoritmo denominado **MLkNN_RST** (*Multi-Label k-Nearest Neighbor and Rough Set Theory*). Este algoritmo utiliza la metaheurística PSO (James Kennedy and Eberhart, 1995) para la selección de atributos, asignando un peso por cada atributo y utiliza como función de calidad la Medida de la Calidad de la Similaridad (Filiberto, 2011).

2.2.1 Selección de atributos utilizando PSO

En las tareas asociadas con el análisis de datos aparece como un problema la selección de atributos (Ruiz and Aguilar, 2005).

El problema de selección de atributos consiste en encontrar el subconjunto de atributos que mejor describe los objetos del dominio. Existen diferentes enfoques y técnicas para seleccionar atributos relevantes, tales como el enfoque lógico combinatorio, el estadístico y métodos basados en búsqueda heurística (Ruiz, 2010). El resultado de la selección puede ser binario {0,1}, en el sentido de que cada atributo es seleccionado o no, o puede dar como

resultado el grado de relevancia del atributo (peso), usualmente en el intervalo [0,1] (Filiberto, 2011).

El cálculo del peso de los rasgos consiste en encontrar el conjunto de pesos $w=\{w(1), w(2),\ldots,w(r)\}$, donde r es la cantidad de rasgos en el conjunto. Para encontrar el conjunto w en (Filiberto, 2011) se propone usar un método de búsqueda heurística tal como PSO (J. Kennedy and Eberhart, 1995), pues esta metaheurística ha mostrado un buen desempeño en los problemas de optimización con valores continuos (Parsopoulos and Vrahatis, 2002; Reyes-Sierra and Coello, 2006).

PSO es una técnica de optimización basada en el comportamiento de una población como son los enjambres de abejas, cardúmenes de peces o bandadas de aves. Estas poblaciones son identificadas como partículas, cada partícula cuenta con una posición y una velocidad en el espacio de búsqueda, además de una función basada en su calidad.

El funcionamiento de esta metaheurística está basado en que cada partícula conoce la posición de las demás y de esta forma adecúa su posición y su velocidad hacia la mejor posición conocida hasta ese instante. La población se mueve por el espacio de búsqueda de un problema n-dimensional y evalúa las posiciones que alcanza cada partícula, según la función a optimizar, llevando un registro de los mejores puntos que se alcanzan.

La velocidad de una partícula i se denota por VX_i, este es un vector que almacena cada una de las velocidades que tiene la partícula en cada dimensión, esta velocidad es añadida a la posición de la partícula, para mover la partícula desde un tiempo t-1 a un tiempo t (Filiberto, 2011). La posición de una partícula i cualquiera se denota por PX_i, donde PX_i es un vector que almacena el valor que tiene la partícula en cada una de las dimensiones que comprende el espacio de búsqueda.

La velocidad con que se va a mover la partícula está condicionada por la suma de tres aspectos, la velocidad anterior de la partícula (inercia), la diferencia entre la mejor posición encontrada por la partícula y la posición actual (aprendizaje propio), y la diferencia entre la mejor posición alcanzada por la nube de partículas y la posición actual de la partícula (aprendizaje del grupo).

Destacar que mientras mayor sean los valores de la cantidad de iteraciones y la cantidad de partículas mayor es la posibilidad de encontrar el óptimo, pero aumenta el costo computacional.

Al terminar el proceso de exploración del espacio de búsqueda, se obtiene un vector de pesos *w* con *r* componentes. La calidad de la partícula se calcula usando la Medida de la Calidad de la Similaridad descrita en los siguientes epígrafes.

2.2.2 Medida de Calidad de la Similaridad adaptada a la problemática de la MLC

En esta investigación se adapta la Medida de la Calidad de la Similaridad propuesta en (Filiberto *et al.*, 2010) a la problemática de la MLC (ver Ecuación 2.9).

Esta propuesta básicamente mide el grado en que dos objetos son similares teniendo en cuenta tanto sus rasgos predictores como su rasgo decisión el cual representa el grado en el cual la similaridad entre los objetos, usando los rasgos predictores, es equivalente a la similaridad que se obtiene de acuerdo al rasgo de decisión d.

$$\theta(\mathrm{DS}) = \sum_{i=1}^{n} \varphi(x)/n \quad (2.9)$$

Donde $\varphi(x)$ (ver Ecuación 2.10) se calcula para cada instancia X_i de X, indicando la cantidad de objetos que están en la intersección de los conjuntos $N_1(X)$ y $N_2(X)$, estos denotan la cantidad de objetos que se encuentran en la vecindad X_i, es decir, que son similares a él según los rasgos predictores (o de condición) y los que son similares a él según los rasgos objetivos (o de decisión) respectivamente.

$$\varphi(x) = \frac{|N_1(x) \cap N_2(x)|}{0.5 * |N_1(x)| + 0.5 * |N_2(x)|} \quad (2.10)$$

Para construir los conjuntos de objetos N_1 y N_2 se usan las relaciones de similaridad R_1 y R_2 propuestas en (Filiberto, 2011) (ver Ecuaciones 2.11 y 2.12).

$$N_1(x) = \{y \in U : xR_1y\} \quad (2.11)$$

$$N_2(x) = \{y \in U : xR_2y\} \quad (2.12)$$

El problema de encontrar una relación de similaridad adecuada para la RST extendida consiste en encontrar las relaciones R_1 y R_2 que logran la mayor semejanza posible entre los

conjuntos N_1 y N_2, esto es $N_1(x) \approx N_2(x)$, donde el símbolo (≈) denota la mayor similaridad posible entre los conjuntos $N_1(x)$ y $N_2(x)$ dado los umbrales de semejanza ε_1 y ε_2, estos toman valores entre 0 y 1 (ver Ecuaciones 2.13 y 2.14).

$$xR_1y \text{ si y solo si } F_1(x,y) \geq \varepsilon_1 \quad (2.13)$$

$$xR_2y \text{ si y solo si } F_2(x,y) \geq \varepsilon_2 \quad (2.14)$$

Siendo F_1 y F_2 funciones de semejanza para calcular el grado de similaridad entre pares de objetos en el universo U. A partir de esto se definen las funciones F_1 y F_2 según las ecuaciones (2.15) y (2.17) respectivamente.

$$F_1(x,y) = \sum_{j=1}^{m} w_j * \partial_j(x,y) \quad (2.15)$$

Donde m es la cantidad de rasgos, x_i y y_i son los valores del rasgo i en los objetos x e y respectivamente, w_i el vector de pesos, y $\partial_i(x_i, y_i)$ se define según la Ecuación (2.16).

$$\partial_j(x, y) = \begin{cases} 1 - \dfrac{|x-y|}{Max(a_j) - Min(a_j)} & \text{si j es continuo} \\ 1 \quad \text{si j es discreto y } x = y \\ 0 \quad \text{si j es discreto y } x \neq y \end{cases} \quad (2.16)$$

$$F_2(x,y) = \partial_d\big(x(d), y(d)\big) \quad (2.17)$$

Donde $\partial_d\big(x(d), y(d)\big)$ se define utilizando relaciones de equivalencia (ver Ecuación (2.18)), la distancia de Jamming (ver Ecuación (2.19)) y la distancia de Jaccard (ver Ecuación (2.20)).

$$\text{equivalencia}\big(x(d), y(d)\big) = \{(x,y) \,/\, x(d) = y(d)\} \quad (2.18)$$

$$\text{distanciaHamming}\big(x(d), y(d)\big) = |x(d) \cup y(d)| - |x(d) \cap y(d)| \quad (2.19)$$

$$\text{distanciaJaccard}\big(x(d), y(d)\big) = \frac{|x(d) \cup y(d)| - |x(d) \cap y(d)|}{|x(d) \cup y(d)|} \quad (2.20)$$

Con el objetivo de seleccionar los k vecinos más cercanos se usa la distancia euclideana ponderada (ver Ecuación (2.21)).

$$d(x, O_i) = \sqrt{\sum_{j=1}^{m} w_j * \partial\left(a_j(P), a_j(O_i)\right)^2} \quad (2.21)$$

$$\partial\left(a_j(P), a_j(E_j)\right) = \begin{cases} \dfrac{\left|a_j(P) - a_j(E_j)\right|}{Max(a_j) - Min(a_j)} & \text{si j es un rasgo continuo} \\ 1 \quad \text{si j es discreto y } a_j(P) = a_j(E_j) \\ 0 \quad \text{si j es discreto y } a_j(P) \neq a_j(E_j) \end{cases}$$

El siguiente pseudocódigo formaliza la propuesta. Este método requiere especificar el conjunto de datos, la instancia de prueba, el número de iteraciones y los individuos.

```
Entrada: conjunto de datos, nueva instancia
Salida: conjunto de etiquetas

P1: Determinar la importancia de los atributos
  P11: Inicializar parámetros PSO (num_Iteraciones,
  num_Particulas)
    w= {w (1), w (2), …, w(n)}. (Generación aleatoria de las
    partículas, teniendo en cuenta posición y velocidad.)
  P12: Determinar la mejor partícula.
  for i=1 hasta num_Iteraciones
    for i=1 hasta num_Particulas
      Actualizar velocidad y posición.
      Calcular óptimo local utilizando medida de la calidad
      de la similaridad definida en (2.9)
    end for
  end for
P2: Calcular las probabilidades ídem a los pasos MLkNN.
  P21: Calcular Probabilidad a priori.
  P22: Calcular Probabilidad a posteriori.
P3: Seleccionar las k instancias más cercanas de acuerdo con
(2.21)
```

Figura 1. Algoritmo MLkNN_RST

2.3 Descripción de la segunda propuesta

Se propone también el algoritmo **MLkNN_FRST** (*Multi-Label k-Nearest Neighbor and Fuzzy-Rough Set Theory*). Este algoritmo es muy similar a MLkNN_RST, la diferencia radica en que utiliza como función de calidad la Medida de la Calidad de la Similaridad Borrosa propuesta en (Coello *et al.*, 2017) pero adaptada a la problemática de la MLC.

2.3.1 Medida de la Calidad de la Similaridad Borrosa adaptada a la problemática de la MLC

La Medida de la Calidad de la Similaridad Borrosa utiliza una relación borrosa binaria, que cuantifica la fortaleza de la relación de semejanza entre dos objetos en un rango de [0,1]. La relación borrosa está caracterizada por una función de pertenencia que se define por una función de semejanza basada en la Teoría de los Conjuntos Aproximados Extendida.

En un conjunto clásico se asigna el valor 0 o 1 a cada elemento para indicar la pertenencia o no a dicho conjunto. Esta función puede generalizarse de forma que los valores asignados a los elementos del conjunto caigan en un rango particular, y con ello indicar el grado de pertenencia de los elementos al conjunto en cuestión (Ramírez and Laguna., 2012). A dicha función se le denomina "función de pertenencia" y el conjunto por ella definida "conjunto borroso". Esta función de pertenencia es definida por μA para un conjunto borroso *A*, determinado por el rango de los números entre [0,1], definido de la siguiente manera: $\mu A = X \rightarrow [0.1]$ (Coello *et al.*, 2017).

Al igual que en la Medida de la Calidad de la Similaridad, la Medida de la Calidad de la Similaridad Borrosa $\theta(DS)$ (ver Ecuación 2.22) representa el grado de similaridad entre los objetos de un sistema de decisión heterogéneo.

$$\theta(\mathrm{DS}) = \left\{ \frac{\sum_{i=1}^{n} \varphi(x)}{n} \right\} \quad (2.22)$$

Donde

$$\varphi(x) = \frac{\sum_{i=1}^{n} [1 - |\mu R_1(x_i) - \mu R_2(x_i)|]}{n} \quad (2.23)$$

Las relaciones borrosas R_1 y R_2 son propuestas en (Coello *et al.*, 2017) y se utilizan para definir los conjuntos borrosos N_1 y N_2 respectivamente (ver Ecuaciones 2.24 y 2.25)

$$N_1(x) = \{(y, \mu R_1(x,y)) \text{ para todo } y \in U\} \quad (2.24)$$

$$N_2(x) = \{(y, \mu R_2(x,y)) \text{ para todo } y \in U\} \quad (2.25)$$

Entonces el problema de encontrar una relación de semejanza adecuada para la RST extendida consiste en encontrar las relaciones R_1 y R_2 que logren mayor similaridad posible

entre los conjuntos borrosos $N_1(x)$ y $N_2(x)$ (Coello *et al.*, 2017). Las relaciones R_1 (ver Ecuación 2.26) y R_2 (ver Ecuación 2.27) están definidas por las funciones de semejanza F_1 y F_2 definidas en el epígrafe anterior (ver Ecuaciones 2.15 y 2.17).

$$\mathrm{xR_1y} = \mathrm{F_1(x,y)} \quad (2.26)$$

$$\mathrm{xR_2y} = \mathrm{F_2(x,y)} \quad (2.27)$$

2.4 Implementación en Mulan de la nueva propuesta

Mulan es una biblioteca Java de código abierto destinada a trabajar con conjuntos de datos multi-etiqueta, o sea, que cada elemento del conjunto de datos puede ser miembro de varias categorías, etiquetas o clases.

Esta biblioteca consta con una variedad de algoritmos destinados a las principales tareas del aprendizaje multi-etiqueta, así como para su evaluación.

Al ser una biblioteca no consta con una interfaz de usuario (GUI), solo cuenta con una interfaz de programación de aplicaciones (API). A partir de esta se incorporan los algoritmos **MLkNN_RST** y **MLkNN_FRST**, en la Figura 2 se muestra una imagen de la inclusión de estos algoritmos en Mulan.

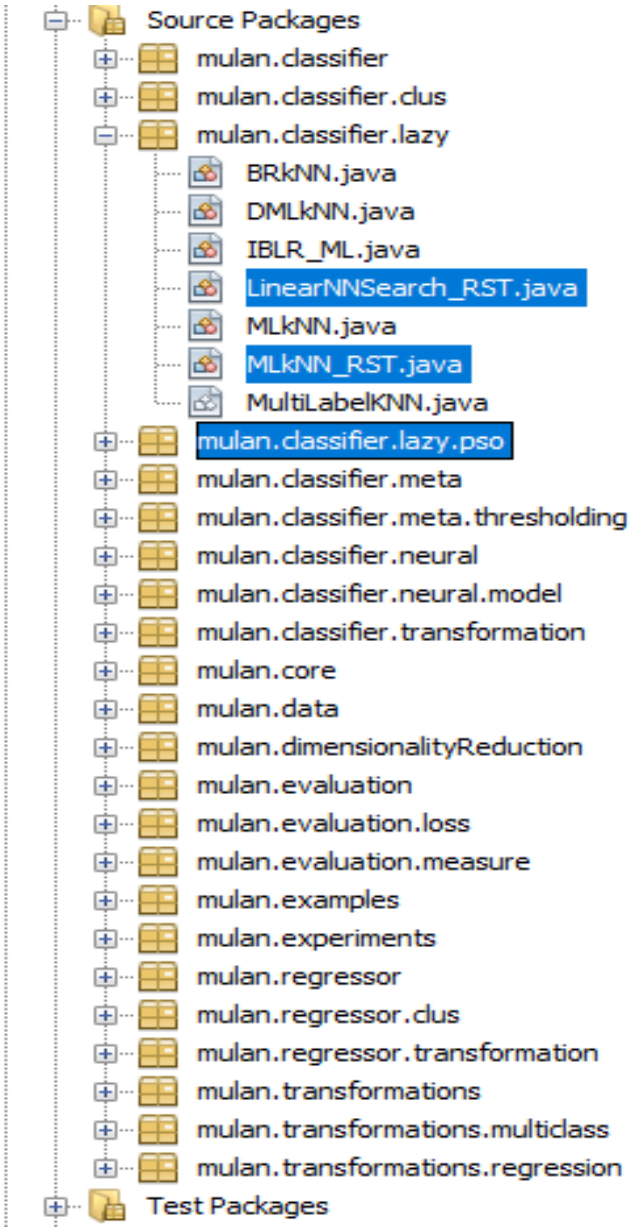

Figura 2. Incorporación de las clases necesaria a MULAN

Para su incorporación fue necesario extender de la clase *MultiLabelKNN*, existente en el paquete *mulan.classifier.lazy* y modificar los métodos *buildInternal* y *makePredictionInternal*.

Dentro de este paquete también se anexa la clase *LinearNNSearch_RST*, y se modifica el método *kNearestNeighbours* a partir de la nueva propuesta para la recuperación de las *k* instancias más cercanas.

Además, se crea el paquete *mulan.classifier.lazy.pso* con las clases *Pso* y *Particle* para el trabajo con la metaheurística PSO.

2.5 Conclusiones parciales del capítulo

Se proponen dos adaptaciones (**MLkNN_RST** y **MLkNN_FRST**) del algoritmo ML-kNN basadas en extensiones de la Teoría de los Conjuntos Aproximados Extendida para resolver problemas de clasificación de multi-etiquetas en conjuntos de datos con presencia de inconsistencia.

Estas modificaciones se basan en modificar la recuperación de las *k* instancias más cercanas, asignando un peso a cada rasgo. Estos pesos son calculados a partir de la metaheurística PSO y utilizando como función de calidad la Medida de la Calidad de la Similaridad y la Medida de la Calidad de la Similaridad Borrosa propuestas por (Filiberto, 2011; Coello *et al.*, 2017) y adaptadas a la problemática de la MLC.

Ambos algoritmos se incorporaron en la biblioteca Mulan.

Capítulo 3 Evaluación del modelo propuesto

En este capítulo se muestran los experimentos realizados a los algoritmos **MLkNN_RST** y **MLkNN_FRST** incorporados a la biblioteca Mulan, a partir de los conjuntos de datos multi-etiqueta más utilizados en la literatura.

3.1 Caracterización de los conjuntos de datos multi-etiquetas utilizadas en la experimentación

Los conjuntos de datos multi-etiqueta utilizados pertenecen a gran cantidad de dominios, entre ellos se destacan: clasificación de música y escenas, clasificación de proteínas y genes, diagnóstico médico, clasificación de textos, entre otros.

En la Tabla 1 se presenta la caracterización de los mismos, atendiendo a la cantidad de instancias, atributos y etiquetas.

En la última columna de la tabla se muestra el valor de inconsistencia obtenido por cada conjunto de dato a partir de la Ecuación (3.1). Esta medida es propuesta en (Filiberto, 2011) y se adaptó para tratar con conjuntos de datos multi-etiquetas; valores cercanos a 0 muestran una alta inconsistencia, mientras que valores cercanos a 1 muestran una alta consistencia en el conjunto de datos.

$$I(d) = \frac{\sum_{i=1}^{m} \left| R_{*}^{'}(X_i) \right|}{\sum_{i=1}^{m} |X_i|} \quad (3.1)$$

Donde,

$R_{*}^{'}(X_i)$: es la aproximación inferior para cada etiqueta.

Nombre	Cantidad Instancias	Cantidad Atributos	Cantidad Etiquetas	Consistencia
emotions	593	72	6	0.983
scene	2407	294	6	0.025
flags	194	19	7	0.994
yeast	2417	103	14	0.000
cal500	502	68	174	0.380
foodtruck	407	33	12	1.000

stackex_coffee	225	1886	123	0.072
stackex_chess	1675	812	227	0.357

Tabla 1.Caracterización de los conjuntos de datos.

3.2 Configuración de parámetros para los algoritmos MLkNN_RST y MLkNN_FRST

En la experimentación con la metaheurística PSO se utilizaron los siguientes valores: entre 5 y 20 partículas, entre 10 y 100 ciclos, y umbrales de similitud entre 0.75 y 0.95. Se realizó un estudio con varias combinaciones de estos parámetros donde se demostró que al aumentar la cantidad de partículas y ciclos la diferencias en los resultados en función de su eficacia no son significativas, sin embargo, la complejidad computacional de ambos algoritmos aumenta a medida que los valores de estos parámetros crecen.

Se utilizaron los mismos parámetros por defecto propuesto para MLkNN en la biblioteca Mulan tanto para los dos algoritmos propuestos como para los algoritmos utilizados en la comparación durante el estudio experimental.

3.3 Estudio experimental

Para el estudio experimental fueron utilizadas las métricas: *Haming Loss*, *Example-Based Precisión*, *Average Precisión* y *Example-Based Accuracy* propuesta en (Schapire and Singer, 2000).

Los dos métodos propuestos se compararon con el método ML-kNN y con otras extensiones del mismo como: BRkNN, DMLkNN y IBLR-ML, propuesta por (M.-L. Zhang and Zhou, 2007; Liu *et al.*, 2010; Zhang, Zhou and Member, 2015) e incorporados a la biblioteca Mulan.

3.3.1 Resultados de la comparación utilizando *Hamming Loss*

En la Tabla 2 se muestran los resultados alcanzados por la métrica *Hamming Loss* para cada conjunto de datos. En **negrita** se destacan los mejores valores obtenidos en cada caso.

	emotions	scene	flags	yeast	cal500	food	coffee	chess
MLkNN_RST	0.1887	0.0826	0.26	**0.1922**	0.1385	0.1569	0.0162	0.105
MLkNN_FRST	0.1887	**0.08**	**0.2502**	0.1928	**0.1380**	**0.1517**	0.0162	**0.0104**
ML-kNN	0.1951	0.0862	0.2536	0.1933	0.1388	0.1561	0.0165	0.105
BRkNN	0.1934	0.0920	0.2748	0.1952	0.1425	0.1557	0.0161	0.0105

DMLkNN	0.1940	0.0863	0.2572	0.1929	0.1381	0.1569	**0.0161**	0.0105
IBLR-ML	**0.1883**	0.0834	0.2703	0.1934	0.2309	0.1519	0.0350	0.0365

Tabla 2. Resultados alcanzados utilizando la métrica *Hamming Loss*.

En la mayoría de los casos el algoritmo **MLkNN_FRST** obtiene los mejores resultados, en particular en los conjuntos de datos donde hay una alta presencia de inconsistencia. Además, se destaca la superioridad del algoritmo **MLkNN_FRST** con respecto al algoritmo **MLkNN_RST**. La Figura 3 ratifica estos resultados, en la misma se compara el algoritmo **MLkNN_FRST** con ML-kNN.

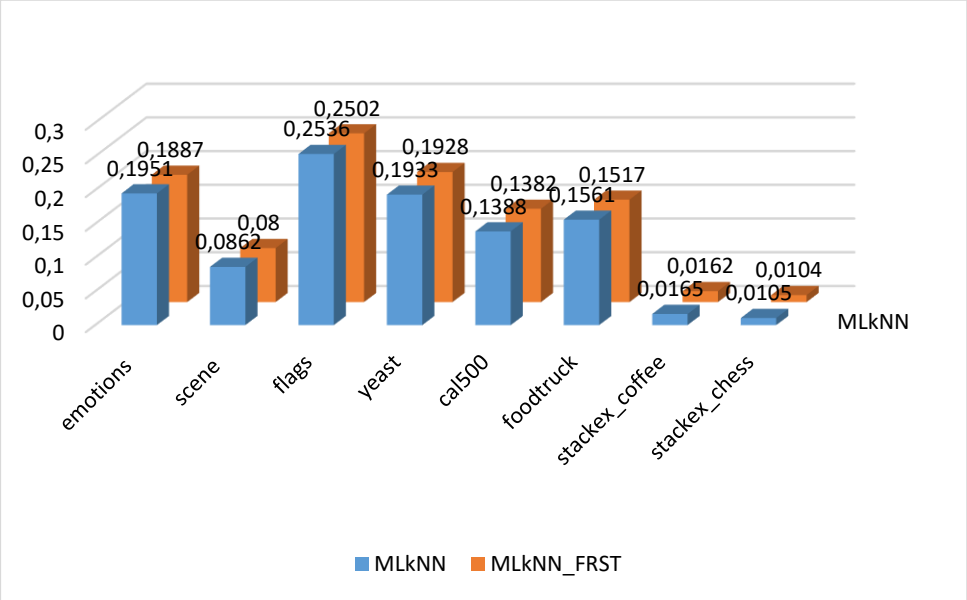

Figure 3. Resultados de la comparación del algoritmo MLkNN_FRST y MLkNN utilizando *Hamming Loss.*

Con el fin de examinar la existencia de diferencias significativas en el rendimiento de estos algoritmos, el siguiente paso es determinar si la superioridad del algoritmo **MLkNN_FRST** con respecto a MLkNN es estadísticamente significativa. Para ellos se aplica la prueba de rango con signo de *Wilcoxon* (Wilcoxon, 1945).

Como resultado de esta prueba se obtiene (*p-value*= 0.011719), rechazándose así la hipótesis nula, utilizando un 95% como nivel de confianza. Por lo que se confirma la existencia de diferencias significativas entre ambos algoritmos.

3.3.2 Resultados de la comparación utilizando *Example-Based Precision*

En la Tabla 3 se muestran los resultados alcanzados por la métrica *Example-Based Precision* para cada conjunto de datos. En **negrita** se destacan los mejores valores obtenidos en cada caso.

	emotions	scene	flags	yeast	cal500	food	coffee	chess
MLkNN_RST	0.6942	**0.7208**	0.7034	0.7228	0.6072	0.6843	**0.0045**	0.1261
MLkNN_FRST	**0.6956**	**0.7208**	**0.7319**	**0.7253**	0.6118	0.6947	**0.0045**	**0.1369**
ML-kNN	0.6883	0.6934	0.7281	0.7198	0.6020	**0.6950**	**0.0045**	0.0776
BRkNN	0.6725	0.6429	0.7317	0.7069	0.5720	0.6550	0.0000	0.0456
DMLkNN	0.6896	0.6992	0.7255	0.7222	**0.6131**	0.6890	0.0000	0.0639
IBLR-ML	0.6705	0.7156	0.7005	0.7139	0.2925	0.6800	0.0275	0.0759

Tabla 3. Resultados alcanzados utilizando la métrica *Example-Based Precision*.

En la mayoría de los casos el algoritmo **MLkNN_FRST** obtiene los mejores resultados, en particular en los conjuntos de datos donde hay una alta presencia de inconsistencia. Además, se destaca la superioridad del algoritmo **MLkNN_FRST** con respecto al algoritmo **MLkNN_RST**. La Figura 4 ratifica estos resultados, en la misma se compara el algoritmo **MLkNN_FRST** con ML-kNN.

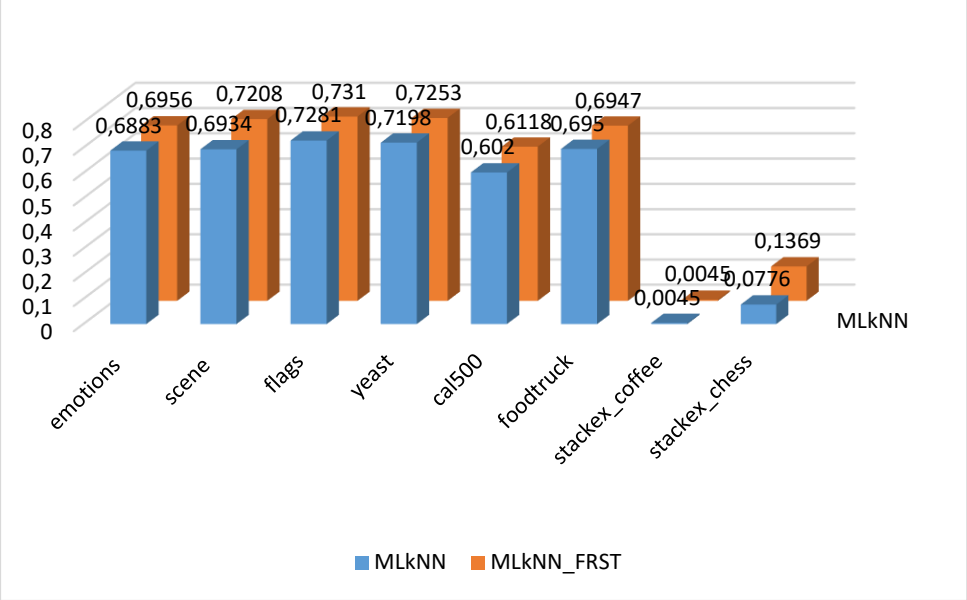

Figure 4. Resultados de la comparación del algoritmo MLkNN_FRST y MLkNN utilizando Example-Based Precision

Con el fin de examinar la existencia de diferencias significativas en el rendimiento de estos algoritmos, el siguiente paso es determinar si la superioridad del algoritmo MLkNN_FRST con respecto a MLkNN es estadísticamente significativa. Para ellos se aplica la prueba de rango con signo de *Wilcoxon* (Wilcoxon, 1945).

Como resultado de esta prueba se obtiene (*p-value*= 0,027992), rechazándose así la hipótesis nula, utilizando un 95% como nivel de confianza. Por lo que se confirma la existencia de diferencias significativas entre ambos algoritmos.

3.3.3 Resultados de la comparación utilizando *Example-Based Accuracy*

En la Tabla 4 se muestran los resultados alcanzados por la métrica *Example-Based Accuracy* para cada conjunto de datos. En **negrita** se destacan los mejores valores obtenidos en cada caso.

	emotions	scene	flags	yeast	cal500	food	coffee	chess
MLkNN_RST	0.5481	0.6871	0.5788	0.5141	0.1954	0.4165	0.0011	0.0589
MLkNN_FRST	**0.5612**	**0.6937**	**0.6182**	0.5153	0.1951	0.4306	0.0023	**0.0632**
ML-kNN	0.5326	0.667	0.6104	0.5162	**0.1972**	0.4308	0.0023	0.0357
BRkNN	0.5149	0.6198	0.5636	0.5002	0.1856	0.4177	0.0000	0.0206
DMLkNN	0.5431	0.6724	0.6065	**0.5182**	0.1929	0.4261	0.0000	0.0292
IBLR-ML	0.5518	0.6902	0.5883	0.5242	0.1949	**0.4416**	**0.0224**	0.0558

Tabla 4. Resultados alcanzados utilizando la métrica *Example-Based Accuracy*

En esta medida el algoritmo **MLkNN_FRST** obtiene buenos resultados, aunque no con tanta superioridad como en el caso de las anteriores, en particular en los conjuntos de datos donde hay una alta presencia de inconsistencia. La Figura 5 ratifica muestra los resultados, en la misma se compara el algoritmo **MLkNN_FRST** con ML-kNN.

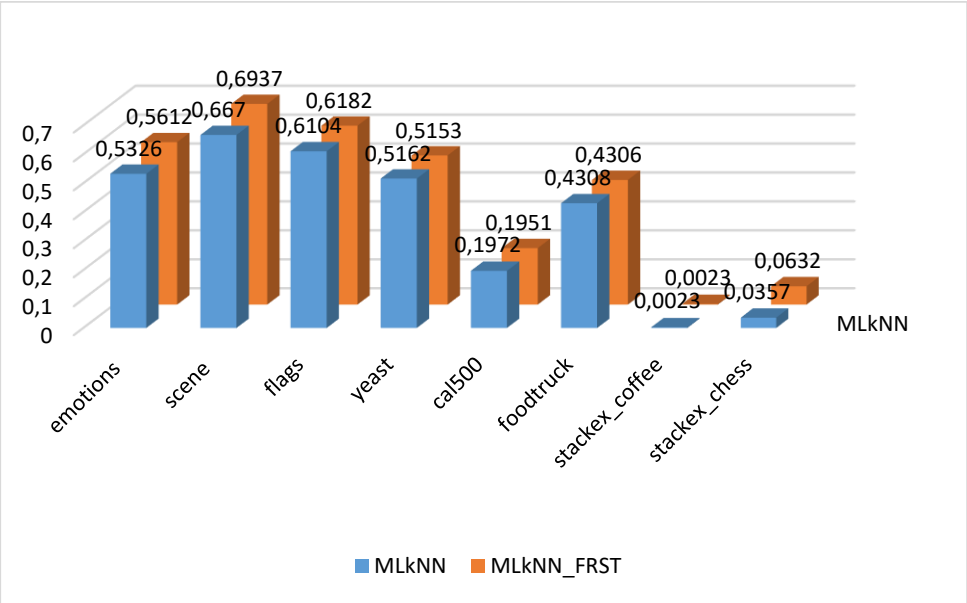

Figura 5. Resultados de la comparación del algoritmo MLkNN_FRST y MLkNN utilizando *Example-Based Accuracy*

Con el fin de examinar la existencia de diferencias significativas en el rendimiento de estos algoritmos, el siguiente paso es determinar si la superioridad del algoritmo MLkNN_FRST con respecto a MLkNN es estadísticamente significativa. Para ellos se aplica la prueba de rango con signo de *Wilcoxon* (Wilcoxon, 1945).

Como resultado de esta prueba se obtiene (*p-value*= 0,176296), aceptándose así la hipótesis nula, utilizando un 95% como nivel de confianza. Por lo que no hay diferencias significativas entre ambos algoritmos.

3.3.4 Resultados de la comparación utilizando *Average Precision*

En la Tabla 5 se muestran los resultados alcanzados por la métrica *Average Precision* para cada conjunto de datos. En **negrita** se destacan los mejores valores obtenidos en cada caso.

	emotions	scene	flags	yeast	cal500	food	coffee	chess
MLkNN_RST	0.8101	0.8739	0.8220	0.7665	0.4941	**0.7432**	**0.2498**	0.2987
MLkNN_FRST	**0.8151**	**0.876**	**0.8327**	0.7679	0.4922	0.7427	0.2463	**0.3009**
ML-kNN	0.7965	0.8662	0.8255	0.7658	0.4942	0.7413	0.2399	0.2925
BRkNN	0.8037	0.8496	0.8280	0.7599	0.4589	0.7033	0.1844	0.2497
DMLkNN	0.7965	0.8673	0.8252	0.7661	**0.4987**	0.7420	0.2404	0.2822
IBLR-ML	0.8126	0.8673	0.8157	**0.7687**	0.2715	0.7316	0.1473	0.1326

Tabla 5. Resultados alcanzados utilizando la métrica *average precision*

En la mayoría de los casos el algoritmo **MLkNN_FRST** obtiene los mejores resultados, en particular en los conjuntos de datos donde hay una alta presencia de inconsistencia. Además, se destaca la superioridad del algoritmo **MLkNN_FRST** con respecto al algoritmo **MLkNN_RST**. La Figura 6 ratifica estos resultados, en la misma se compara el algoritmo **MLkNN_FRST** con ML-kNN.

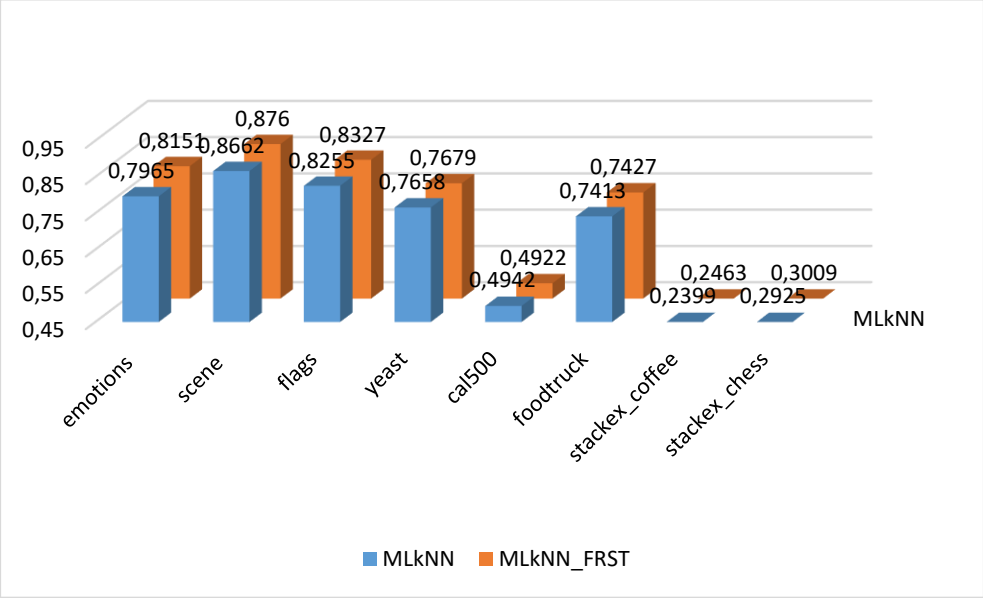

Figure 6. Resultados de la comparación del algoritmo MLkNN_FRST y MLkNN utilizando *Average Precision*

Con el fin de examinar la existencia de diferencias significativas en el rendimiento de estos algoritmos, el siguiente paso es determinar si la superioridad del algoritmo MLkNN_FRST con respecto a MLkNN es estadísticamente significativa. Para ellos se aplica la prueba de rango con signo de *Wilcoxon* (Wilcoxon, 1945).

Como resultado de esta prueba se obtiene (*p-value*= 0,025062), rechazándose así la hipótesis nula, utilizando un 95% como nivel de confianza. Por lo que se confirma la existencia de diferencias significativas entre ambos algoritmos.

3.4 Conclusiones parciales

A partir de los conjuntos de datos y métricas seleccionadas el análisis estadístico demuestra que:

- El algoritmo MLkNN_RST obtiene resultados similares al algoritmo MLkNN.
- La superioridad del algoritmo MLkNN_FRST respecto a los algoritmos MLkNN_RST, y MLkNN, así como a otras extensiones del mismo existentes en Mulan; en particular en datos con alta presencia de inconsistencia.

Conclusiones

Se proponen dos adaptaciones del algoritmo MLkNN basadas en extensiones de la Teoría de los Conjuntos Aproximados para resolver problemas de clasificación de multi-etiquetas en conjuntos de datos con presencia de inconsistencia.

Se implementaron estas dos adaptaciones en la biblioteca Mulan, MLkNN_RST y MLkNN_FRST, utilizando las facilidades de la misma para la incorporación de nuevos algoritmos, así como, para su evaluación. Las nuevas adiciones permitirán a los profesores del laboratorio de Inteligencia Artificial y a la comunidad científica del Aprendizaje Automatizado utilizar estos métodos como referentes en esta área.

Se demuestra la superioridad de la modificación propuesta a ML-kNN usando la Medida de la Calidad de la Similaridad.

La propuesta usando la Medida de la Calidad de la Similaridad Borrosa elimina la dependencia al umbral de similitud y tiene un mejor comportamiento con respecto al algoritmo ML-kNN y a otras adaptaciones existentes del mismo en la biblioteca Mulan.

Recomendaciones

Los resultados alcanzados, así como la experimentación realizada permitió formular nuevas tareas de investigación, las cuales se presentan como recomendaciones de este trabajo:

- Analizar el comportamiento de los dos métodos propuestos a partir de otras funciones de distancias como, *Heterogeneous Manhattan-Overlap Metric* (HMOM) y *Heterogeneous Value Difference Metric* (HVDM).
- Evaluar la paralelización de los algoritmos propuestos con el fin de mejorar la complejidad computacional de los mismos.

Referencias

Agrawal, J. *et al.* (2014) ‘An Investigation of Fuzzy PSO and Fuzzy SVD Based RBF Neural Network for Multi-lab el Classification.’

AHA, M., D.K., D. W. and ALBERT (1991) ‘Instance-based learning algorithms’, *Machine Learning*, 6, pp. 37–66.

Akbarnejad, A. and Baghshah, M. S. (2017) ‘A probabilistic multi-label classifier with missing and noisy labels handling capability’, *Pattern Recognition Letters*. Elsevier B.V., 89, pp. 18–24. doi: 10.1016/j.patrec.2017.01.022.

Ávila Jiménez, J. L. (2013) ‘Modelos de Aprendizaje Basados en Programación Genética para Clasificación Multi-Etiqueta’.

Bi, W. and Kwok, J. T. (2011) ‘Multi-label classification on tree-and dag-structured hierarchies’, in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 17–24.

Bielza, C., Li, G. and Larrañaga, P. (2011) ‘Multi-dimensional classification with Bayesian networks’, *International Journal of Approximate Reasoning*, 52, pp. 705–727.

Botero, J. F. and Palomeque, L. A. (2014) ‘El OVA Como Estrategia Para La Enseñanza Aprendizaje De La Cinética Química’, pp. 737–740.

Boutell, M. *et al.* (2004) ‘Learning multilabel scene classification’, 37, pp. 1757– 1771.

Bratton, D. and Kennedy, J. (2007) ‘Defining a Standard for Particle Swarm Optimization.’

Briggs, F. *et al.* (2012) ‘Acoustic classification of multiple simultaneous bird species: a multi-instance multi-label approach.’, *J. Acoust. Soc. Am.*, 131(6), pp. 4640–4650.

Cakir, E. *et al.* (2015) ‘Polyphonic sound event detection using multi label deep neural networks’, in *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. doi: 10.1109/IJCNN.2015.7280624.

Cárdenas, B., Caballero, Y. and Bello, R. (2007) ‘La Teoría de los Conjuntos Aproximados y las Técnicas de Boostrap para la Edición de el Pronóstico Meteorológico’, 4(3).

Chan, A. and Freitas, A. A. (2006) ‘A new ant colony algorithm for multilabel classification with applications in bioinfomatics’, *GECCO ’06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp. 27–34.

Charte, F. *et al.* (2015) ‘QUINTA: A question tagging assistant to improve the answering ratio in electronic forums’, in *Proceedings of IEEE International Conference on Computer as a Tool, EUROCON’15*, pp. 1–6.

Cheng, W. and Eyke Hullermeier (2009) ‘Combining Instance-Based Learning and Logistic Regression for Multilabel Classification (Resubmission) ∗ ’.

Cheng, W., Hüllermeier, E. and Dembczynski, K. J. (2010) ‘Bayes optimal multilabel classification via probabilistic classifier chains.’

Cherman, E. A., Metz, J. and Monard, M. (2010) '"A Simple Approach to Incorporate Label Dependency in Multi-label Classification"'.

Chou, K.-C., Wu, Z.-C. and Xiao, X. (2011) 'iLoc-Euk: a multi-label classifier for predicting the subcellular localization of singleplex and multiplex eukaryotic proteins', *PloS one*. Public Library of Science, 6(3), pp. 1–10.

Ciarelli, P. M. and Oliveira, E. (2009) 'An Enhanced Probabilistic Neural Network Approach Applied to Text Classification'.

Clare, A. and King, R. D. (2001) 'Knowledge discovery in multi-label phenotype data'.

Coello, L. *et al.* (2017) 'Construcción de relaciones de similaridad borrosa basada en la medida calidad de la similaridad', 38(2), pp. 132–140.

Comité, F. De, Gilleron, R. and Tommasi, M. (2003) 'Learning Multi-label Alternating Decision Trees from Texts and Data.'

Crammer, K. and Singer, Y. (2003) 'A family of additive online algorithms for category ranking', *The Journal of Machine Learning Research*, 3, pp. 1025–1058.

Dimitrovski, I. *et al.* (2010) '"Detection of Visual Concepts and Annotation of Images Using Ensembles of Trees for Hierarchical Multi-Label Classification"'.

Diplaris, S. *et al.* (2005) 'Protein classification with multiple algorithms', *Advances in Informatics*, pp. 448–456.

Dorigo, M., Caro, G. and Gambardella, L. (1999) 'Ant algorithms for discrete optimization.'

Elisseeff, A. and Weston, J. (2001) 'A Kernel Method for Multi-Labelled Classification', *Advances in Neural Information Processing Systems*, 14, pp. 681–687.

Filiberto, Y. *et al.* (2010) 'A method to build similarity relations into extended Rough Set Theory', in *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pp. 1314–1319.

Filiberto, Y. (2011) 'Métodos de aprendizaje para dominios con datos mezclados basados en la teroía de los conjuntos aproximados extendida'.

Filiberto, Y. and Bello, R. (2010) 'Using PSO and RST to Predict the Resistant Capacity of Connections in Composite Structures.', *In International Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) Springer.*, pp. 359–370.

Freud, Y. and Mason, L. (1999) 'The alternating decision tree learning algorithm.'

Ghamrawi, N. and McCallum, A. (2005) 'Collective multi-label classification', *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 195–200.

Gibaja, E. and Ventura, S. (2014) *Multi-label learning: a review of the state of the art and ongoing research.*

Grodzicki, R., Mańdziuk, J. and Wang, L. (2008) 'Improved multilabel classification with

neural networks.’

Han, J. and Kamber, M. (2006) ‘Data Mining: Concepts and Techniques’.

Jiang, J.-Y., Tsai, S.-C. and Lee, S.-J. (2012) ‘FSKNN: multi-label text categorization based on fuzzy similarity and k nearest neighbors.’, *Expert Systems with Applications*, 39, pp. 2813–2821.

Kennedy, J. and Eberhart, R. (1995) ‘PSO optimization’, in *Proc. IEEE Int. Conf. Neural Networks*, pp. 1941–1948.

Kennedy, J. and Eberhart, R. C. (1995) ‘Particle swarm optimization.’, *In Proceedings of the 1995 IEEE International Conference on Neural Networks*.

Khan, S. and Rauf, A. (2017) ‘Ant colony optimization based hierarchical multi-label classification algorithm’, *Applied Soft Computing Journal*. Elsevier B.V., 55, pp. 462–479. doi: 10.1016/j.asoc.2017.02.021.

Kongsorot, Y. and Horata, P. (2014) ‘Multi-label classification with extreme learning machine.’

Lewis, D. D. *et al.* (2004) ‘RCV1: A new benchmark collection for text categorization research’, *Journal of Machine Learning Research*, 5, pp. 361–397.

Li, J. and Xu, J. (2009) ‘“A Fast Multi-label Classification Algorithm Based on Double Label Support Vector Machine”’.

Li, W., Han, J. and Pei, J. (2001) ‘CMAR: Accurate and efficient classification based on multiple class-association rules’.

Liu, G. *et al.* (2010) ‘Modelling of inquiry diagnosis for coronary heart disease in traditional Chinese medicine by using multi-label learning’.

Nam, J. *et al.* (2014) ‘Large-scale multi-label text classification—revisiting neural networks’, in *Joint european conference on machine learning and knowledge discovery in databases*, pp. 437–452.

Nilsson, N. J. (1996) ‘“Introduction to machine learning”’, *Stanford University*.

Ojeda, F. C. (2015) ‘Nuevos métodos híbridos de computación flexible para clasificación multietiqueta.’

Oliveira, E. *et al.* (2008) ‘Using a Probabilistic Neural Network for a Large Multi-label Problem’, *Neural Networks, 2008. SBRN ’08. 10th Brazilian Symposium on*, pp. 195– 200.

Parpinelli, R. S., Lopes, H. S. and Freitas, A. A. (2002) ‘Data mining with an ant colony optimization algorithm.’

Parsopoulos, K. E. and Vrahatis, M. N. (2002) *Recent approaches to global optimization problems through Particle Swarm Optimization.*

Rak, R., Kurgan, L. and Reformat, M. (2008) ‘A tree-projection-based algorithm for multi-label recurrent-item associative-classification rule generation.’, *Data & Knowledge Engineering*, 64, pp. 171–197.

Ramírez, N. V. and Laguna., M. (2012) ‘La lógica borrosa: conjuntos borrosos, razonamiento aproximado y control borroso.’, *Pistas Educativas*.

Reyes-Sierra, M. and Coello, C. C. (2006) ‘Multi-objective Particle Swarm Optimizers: A survey of the State-of-the-Art.’, *International Journal of Computational Intelligence Research*, pp. 287–308.

Ruiz, J. (2010) ‘Reconocimiento lógico combinatorio de patrones: teoría y aplicaciones.’

Ruiz, R. and Aguilar, J. (2005) ‘Analysis of feature rankings for classification.’, *In Advances in Intelligent Data Analysis VI (IDA 2005), Springer Verlag.*

Russell, S. J. and Norvig, P. (1996) ‘Inteligencia artificial:un enfoque moderno.’, *Prentice Hall Hispanoamericana S.A.*

Schapire, R. E. and Singer, Y. (1999) ‘Improved boosting algorithms using confidencerated predictions.’

Schapire, R. E. and Singer, Y. (2000) ‘BoosTexter: A boosting-based system for text categorization.’, *Machine learning*, 39, pp. 135–168.

Shao, H. *et al.* (2013) ‘Symptom selection for multi-label data of inquiry diagnosis in traditional Chinese medicine’, *Science China Information Sciences*. Springer, 56(5), pp. 1–13.

Spyromitros, E., Tsoumakas, G. and Vlahavas, I. (2008) ‘An empirical study of lazy multilabel classification algorithms’, in *Hellenic conference on artificial intelligence*, pp. 401–406.

Sucar, L. E. *et al.* (2014) ‘Multi-label classification with bayesian network-based chain classifiers.’

Thabtah, F. A. and Cowling, P. I. (2007) ‘A greedy classification algorithm based on association rule’.

Thabtah, F. A., Cowling, P. and Peng, Y. (2004) ‘MMAC: A new multi-class, multi-label associative classification approach’, pp. 217–224.

Thabtah, F. A., Cowling, P. and Peng, Y. (2006) ‘Multiple labels associative classification’, *Knowledge and Information Systems*, 9, pp. 109–129.

Thabtah, F., Cowling, P. and Peng, Y. (2005) ‘MCAR: multi-class classification based on association rule’.

Tsoumakas, G. and Vlahavas, I. (2007) ‘Random k-Labelsets: An Ensemble Method for Multilabel Classification.’, *Proc. 18th European Conf. on Machine Learning, Warsaw, Poland, ECML’07*, 4701, pp. 406–417.

Vallim, R. *et al.* (2008) ‘A new approach for multi-label classification based on default hierarchies and organizational learning.’, *in GECCO ’08: Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*, pp. 2017–2022.

Veloso, A., Goncalves, M. and Zaki, M. (2007) ‘Multi-label lazy associative classification’,

Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2007), pp. 605–612.

Wan, S. P. and Xu, J. H. (2007) '"A multi-label classification algorithm based on triple class support vector machine"'.

Wang, H., Huang, M. and Zhu, X. (2008) 'A Generative Probabilistic Model for Multi-label Classification', *ICDM '08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pp. 628–637.

Wang, J. *et al.* (2014) 'Simplified Constraints RankSVM for Multi-label Classification.'

Wang, J. *et al.* (2016) 'Cnn-rnn: A unified framework for multi-label image classification', in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pp. 2285–2294.

Wilcoxon, F. (1945) 'Individual comparisons by ranking methods Biometrics', *Biometrics*, 1, pp. 80–93.

Witten, I. H., Frank, E. and Hall, M. A. (2011) 'Data Mining: Practical Machine Learning Tools and Techniques.'

Wu, Q. *et al.* (2014) 'MLTREE: A tree-structure-based approach to multilabel learning.'

Xiao, X. *et al.* (2015) 'iDrug-Target: predicting the interactions between drug compounds and target proteins in cellular networking via benchmark dataset optimization approach', *Journal of Biomolecular Structure and Dynamics*. Taylor & Francis, 33(10), pp. 2221–2233. doi: 10.1080/07391102.2014.998710.

Xu, J. (2013) 'Fast multi-lab el core vector machine.'

Younes, Z., Abdallah, F. and DenÃžeux, T. (2008) 'Multi-label classification algorithm derived from k-nearest neighbor rule with label dependencies.', *roceedings of the 16th European Signal Processing Conference*.

Zafra, A. and Ventura, S. (2010) 'G3P-MI: A genetic programming algorithm for multiple instance learning', *Information Sciences*, pp. 4496–4513.

Zhang, M. (2006) 'Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization.'

Zhang, M.-L. (2009) 'Ml-rbf : RBF Neural Networks for Multi-Label Learning', *Neural Processing Letters*, 29, pp. 61–74.

Zhang, M.-L. and Zhou, Z.-H. (2007) 'ML-KNN: A lazy learning approach to multi-label learning', *Pattern recognition*. Elsevier, 40(7), pp. 2038–2048.

Zhang, M. and Zhou, Z. (2007) 'ML-KNN : A lazy learning approach to multi-label learning', 40, pp. 2038–2048. doi: 10.1016/j.patcog.2006.12.019.

Zhang, M., Zhou, Z. and Member, S. (2015) 'Multi-Label Neural Networks with Applications to Functional Genomics and Text Categorization', (November 2006). doi: 10.1109/TKDE.2006.162.

Zong, W. and Huang, G.-B. (2011) ‘Face recognition based on extreme learning machine’, *Neurocomputing*.