UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE INFORMÁTICA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

BARBARA CRISTINA DOS SANTOS

THISSIANY BEATRIZ ALMEIDA

PREVISÃO DA QUANTIDADE DE CLASSES EM CLASSIFICAÇÃO HIERÁRQUICA MULTIRRÓTULO

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2014

BARBARA CRISTINA DOS SANTOS

THISSIANY BEATRIZ ALMEIDA

PREVISÃO DA QUANTIDADE DE CLASSES EM CLASSIFICAÇÃO HIERÁRQUICA MULTIRRÓTULO

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Ciência da Computação, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná.

Orientadora: Profª Drª Helyane B. Borges

PONTA GROSSA

2014

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Ponta Grossa
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Informática
Bacharelado em Ciência da Computação

TERMO DE APROVAÇÃO

PREVISÃO DA QUANTIDADE DE CLASSES EM CLASSIFICAÇÃO HIERÁRQUICA MULTIRRÓTULO

por

BARBARA CRISTINA DOS SANTOS
THISSIANY BEATRIZ ALMEIDA

Este Trabalho de Conclusão de Curso (TCC) foi apresentado em 26 de Maio de 2014 como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Helyane Bronoski Borges
Profª Orientadora

_____
Geraldo Ranthum
Membro titular

_____
Simone de Almeida
Membro titular

- O Termo de Aprovação assinado encontra-se na Coordenação do Curso -

Aos nossos pais e irmãs que, com muito carinho e apoio, não mediram esforços para que chegássemos até aqui.

AGRADECIMENTOS

Primeiramente agradecemos a Deus, que nos permitiu que tudo isso acontecesse, Ele quem nos amparou não apenas nestes anos como universitárias, mas ao longo de nossa vida, nos dando forças para enfrentar os obstáculos encontrados.

Aos nossos pais, pelo apoio e por tudo que sempre fizeram por nós, pela simplicidade, exemplo, e carinho, fundamentais na construção do nosso caráter. Além do grande incentivo nas horas mais difíceis, de desânimo e cansaço.

À nossas irmãs, que nos tomaram como exemplo e que de forma muito especial e carinhosa nos deram forças e coragem, nos apoiando nos momentos de dificuldades.

A toda nossa família, avós, avôs, tios e primos, por compartilharem diversos momentos importantes de nossa vida desde o início dessa caminhada.

À nossa orientadora, Profª Drª Helyane Borges, que acreditou, ouviu pacientemente nossas considerações partilhando conosco suas ideias, conhecimento e experiências. Queremos expressar nosso reconhecimento e admiração pela sua competência profissional, por ser uma profissional extremamente qualificada e pela forma humana como conduziu nossa orientação.

Aos nossos amigos, nossa segunda família. Aos amigos de perto a quem aprendemos a amar e construir laços eternos. Obrigada por todos os momentos em que fomos estudiosos, brincalhões, atletas e cúmplices. Porque em vocês encontramos verdadeiros irmãos. Obrigada pela paciência, pelo sorriso, pelo abraço, pela mão que sempre se estendia quando precisávamos. Esta caminhada não seria a mesma sem vocês. Aos amigos de longe, porque mesmo quando distantes, estavam presentes em nossa vida.

Aos professores por nos proporcionarem o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a nós. A palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão nossos eternos agradecimentos.

RESUMO

SANTOS, Barbara Cristina; ALMEIDA, Thissiany Beatriz. **Previsão da Quantidade de Classes em Classificação Hierárquica Multirrótulo.** 2014. p. 73. Trabalho de Conclusão de Curso Bacharelado em Ciência da Computação - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2014.

Muitos dos problemas de classificação descritos na literatura de Aprendizagem de Máquina dizem respeito à classificação de dados em que cada exemplo é associado a uma classe pertencente a um conjunto finito de classes, todas em um mesmo nível. No entanto, vários problemas de classificação, são de natureza hierárquica, em que classes podem ser subclasses ou superclasses de outras classes. Em muitos problemas hierárquicos, um ou mais exemplos podem ser associados a mais de uma classe simultaneamente. Esses problemas são conhecidos como problemas de classificação hierárquica multirrótulo. Nesse trabalho, foi utilizada a técnica ML-kNN para a predição de problemas multirrótulos, visando determinar o número de classes que podem ser atribuídas a um exemplo.

Palavras-chave: Classificação. Hierárquico. Multirrótulo. Classificação Hierárquica Multirrótulo

ABSTRACT

SANTOS, Barbara Cristina; ALMEIDA, Thissiany Beatriz. **Forecast of Classes numbers in a Hierarchical Multi-label Classification.** 2014. p. 73.Trabalho de Conclusão de Curso Bacharelado em Ciência da Computação - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2014.

Many Machine's Learning classification problems are described in the literature associanting the classification of data with a belonging class to an finite set of classes, all at a same level. However, many classification problems are of hierarchical nature, in which classes may be subclasses or superclasses of other classes. In many hierarchical problems, one or more examples may be associated to more than one class simultaneously. Those problems are known as multi-label hierarchical classification problems. In this paper, the ML-kNN techniques used to address the prediction of multi-label problems, aiming to determine the number of classes that may be assigned to an example.

Keywords: Machine Learning Classification. Hierarchical classification set. Multi-label problem.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE SIGLAS

AM	Aprendizado de Máquina
ARFF	*Attribute-Relation File Format*
BR	*Binary Relevance*
CD	*Critical Distance*
DAG	*Directed Acyclic Graph*
GO	*Gene Ontology*
H-Loss	*Hierarchical Loss Function*
HMC	*Hierarchical Multilabel Classification*
IA	Inteligência Artificial
kNN	*K Nearest Neighbor*
LP	*Label Powerset*
MAP	*Maximum a Posteriori*
ML-kNN	*Multi-Label k-Nearest Neighbor*
RAkEL	*Random k-Labelsets*
SVM	*Support Vector Machines*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Classificação é uma das mais importantes tarefas de Aprendizado de Máquina (AM). Nessa área de pesquisa, um problema de classificação pode ser definido como: dado um conjunto de exemplos de treinamento compostos por pares (T_i, y_i), no qual T_i representa uma tupla de atributos de entrada que descreve um exemplo e y_i, sua classe associada, encontrar uma função que mapeie cada T_i para sua classe associada y_i, tal que $i = 1, 2, \ldots, m$, em que m é o número de exemplos de treinamento (CERRI, 2010).

Muitos dos métodos de AM consideram que cada exemplo está associado a um único rótulo (MITCHELL, 1997). Entretanto, existem vários domínios de aplicação nos quais cada exemplo está associado a mais de um rótulo. Métodos que são capazes de aprender nesses domínios são conhecidos como métodos de aprendizado multirrótulo (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

Existe um grande número de problemas descritos na literatura de AM que dizem respeito a problemas de classificação não hierárquica, onde cada exemplo é associado a uma classe pertencente a um conjunto finito de classes, todas em um mesmo nível. No entanto, existe um grande número de problemas em que uma ou mais classes podem ser divididas/classificadas em subclasses ou agrupadas em superclasses. A esses problemas, dá-se o nome de problemas de classificação hierárquica.

Tais problemas têm por objetivo a classificação de cada novo exemplo em um dos nós-folhas. No entanto, pode ocorrer de o classificador ter uma baixa confiança na classificação em uma das classes desse nível, sendo mais seguro classificá-lo em uma das classes dos níveis mais elevados da hierarquia.

Existem, porém, problemas hierárquicos em que duas ou mais classes podem ser atribuídas a um mesmo exemplo. Esses problemas são muito importantes na prática, principalmente nas áreas de classificação de textos e Bioinformática. Esses problemas, que combinam classificação hierárquica com classificação multirrótulo, são chamados de problemas de classificação hierárquica multirrótulo (CARVALHO; FREITAS, 2007).

A classificação hierárquica é considerada uma área de pesquisa relativamente nova, o que proporciona o interesse de pesquisadores de diferentes

áreas. Alguns exemplos encontram-se na categorização de textos, predição de proteínas, classificação de gêneros musicais e imagens, entre outras. Recentemente, novas abordagens e métodos de classificação têm sido propostos com o intuito de atender às características específicas dos problemas hierárquicos.

A motivação inicial para as pesquisas na área de classificação multirrótulo surgiu com a dificuldade causada por ambiguidades em problemas de categorização de textos (SCHAPIRE; SINGER, 2000).

Este trabalho tem como objetivo adaptar uma técnica utilizada na Inteligência Artificial aplicada a problemas hierárquicos multirrótulos, visando determinar o número de classes que pode ser atribuído a um exemplo real que não pertença à base de treinamento. Através desse trabalho, pretende-se analisar como os algoritmos já existentes fazem a verificação que a classificação dada pelo algoritmo é 100% confiável.

1.1 OBJETIVOS

Os objetivos do trabalho são descritos a seguir. A subseção 1.1.1 descreve o objetivo geral do trabalho. A subseção 1.1.2 relata os objetivos específicos do mesmo.

1.1.1 Objetivo Geral

- Aplicar uma técnica para predizer a quantidade de classes em problemas hierárquicos em técnicas multirrótulos.

1.1.2 Objetivos Específicos

- Estudar classificação hierárquica.
- Estudar sobre problemas multirrótulos.
- Estudar as técnicas utilizadas para predição em problemas multirrótulos.
- Verificar qual técnica se adapta melhor ao problema de classificação hierárquica em problemas multirrótulos.

- Realizar experimentos nas técnicas desenvolvidas/aplicadas.

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho é constituído de seis capítulos. O Capítulo 2 aborda assuntos relevantes para o desenvolvimento do trabalho, como aprendizagem de máquina, classificação de dados, classificação de dados hierárquico, classificação de dados multirrótulo e classificação de dados hierárquico multirrótulo.

O capítulo 3 apresenta a metodologia utilizada, contendo as bases de dados utilizadas nos experimentos, a ferramenta de software utilizada e a metodologia utilizada para a realização dos experimentos.

O capítulo 4 é correspondente aos experimentos e resultados experimentais obtidos com a aplicação da técnica de classificação hierárquica multirrótulo. Neste capítulo encontram-se todos os experimentos realizados com as bases de dados, bem como as configurações para a realização dos mesmos.

O capítulo 5 apresenta as considerações finais desse trabalho, assim como identifica possíveis trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo são destacados os fundamentos teóricos importantes para o desenvolvimento da pesquisa. Na seção 2.1 tem-se conceitos de aprendizagem de máquina. Conceitos de classificação de dados, classificação hierárquica, classificação multirrótulo e classificação hierárquica multirrótulo aparecem nas seções 2.2, 2.3, 2.4 e 2.5, respectivamente.

2.1 APRENDIZAGEM DE MÁQUINA

Aprendizagem de Máquina (AM) é uma subárea da Inteligência Artificial (IA) que mais tem crescido nos últimos anos, sendo proposto continuamente diferentes algoritmos, formas de utilização e extensões dos algoritmos existentes (COSTA, 2008).

Seu objetivo é desenvolver técnicas computacionais que possibilitem que o computador tome decisões baseadas em experiências acumuladas. Dessa maneira, AM é a área que trata a questão de como construir programas que melhorem seu desempenho automaticamente com a experiência (MONARD; BARANAUSKAS, 2003; MITCHELL, 1997).

Um dos principais focos da pesquisa em AM é produzir (induzir) modelos automaticamente a partir de dados, como por exemplo regras e padrões. O processo de indução consiste em realizar inferências gerais a partir de um subconjunto particular de dados. Esse processo de aquisição de conhecimento é denominado de Aprendizagem Indutiva (COSTA, 2008).

Como pode ser observado na hierarquia de aprendizagem mostrada na Figura 1, o processo indutivo de aprendizado pode ser dividido em três categorias: aprendizagem supervisionada, aprendizagem não-supervisionada e aprendizagem semissupervisionada (MONARD; BARANAUSKAS, 2003). A principal diferença entre aprendizagem supervisionada e não-supervisionada diz respeito à forma de como é feito o processo de generalização do conhecimento.

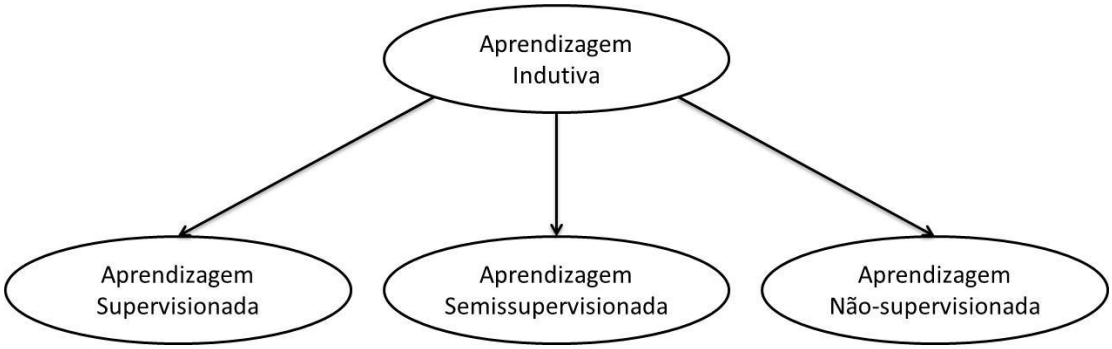

Figura 1 - Hierarquia do aprendizado
Fonte: Monard *et al.* (2003)

Na aprendizagem supervisionada são construídos algoritmos de indução (também chamados de indutores) que realizam inferências a partir de exemplos rotulados. A classificação faz parte desse tipo de aprendizagem de máquina, onde são desenvolvidos algoritmos que realizam induções de classificadores a partir de exemplos previamente classificados. Por sua vez a aprendizagem não-supervisionada tem por objetivo encontrar padrões entre os exemplos de entrada por meio da realização de agrupamentos (*clustering*). Nessa aprendizagem são disponíveis apenas os exemplos de entradas e o indutor não tem informações a respeito da saída esperada (MONARD; BARANAUSKAS, 2003).

Recentemente, a comunidade científica vem estudando uma terceira categoria de aprendizagem, surgida através da junção da aprendizagem supervisionada e não-supervisionada. A essa nova categoria dá-se o nome de aprendizagem semissupervisionada (MATSUBARA *et al.*, 2005). Tal aprendizagem combina dados rotulados e dados não-rotulados durante a fase de treinamento, reduzindo assim, a necessidade de uma grande quantidade de dados rotulados quando somente um pequeno conjunto de exemplos rotulados está disponível.

2.2 CONCEITOS FUNDAMENTAIS DE CLASSIFICAÇÃO

Classificação de dados é o processo de encontrar um modelo que descreva as diferentes classes presentes em um conjunto de dados, ou seja, extrair informações de um conjunto de dados por meio de categorização. Desse modo, um processo de classificação consiste em atribuir rótulos aos dados, de tal forma que

esses rótulos confiram informações aos dados categorizados sob o mesmo rótulo (COSTA, 2008).

Em aprendizagem de máquina, a classificação faz parte de um tipo de aprendizagem denominada de aprendizagem supervisionada, em que são desenvolvidos algoritmos que realizam induções de classificadores a partir de exemplos previamente classificados. Assim, é obtido um classificador utilizando exemplos que contêm a informação da sua saída esperada. Esse classificador é obtido por meio de um algoritmo de indução (indutor), que tem como objetivo fazer com que o classificador seja capaz de classificar corretamente novos exemplos (CERRI, 2010). A Figura 2 ilustra esse processo de classificação.

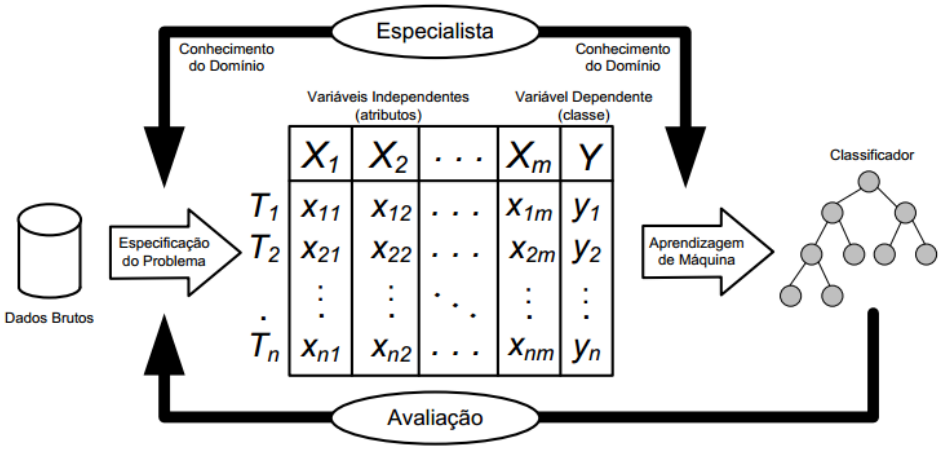

Figura 2 - Processo de classificação
Fonte: Rezende (2005)

Inicialmente os dados brutos devem ser preparados em um conjunto de exemplos para que possam ser processados. Um conjunto de exemplos é composto por valores de atributos, que são características do exemplo, e pelo atributo classe. Na Figura 2 é mostrado o formato padrão de um conjunto de exemplos T com m exemplos e n atributos. A linha i refere-se ao i-ésimo exemplo onde $i = 1, 2, \ldots, n$ e a entrada x_{ij} refere-se ao valor do j-ésimo atributo X_j do exemplo i, onde $j = 1, 2, \ldots, m$ (BORGES, 2012).

Após o processamento dos dados, esse conjunto de exemplos será submetido à entrada do algoritmo de indução para que seja feito o treinamento do classificador. O objetivo do treinamento é encontrar uma função que mapeie cada exemplo T_i com a sua classe y_i correspondente. Segundo Tan *et al.* (2005), o processo de classificação consiste do aprendizado de uma função objetivo f que

mapeia cada conjunto de atributos T_i em uma das classes predefinidas y_i (TAN; STEINBACH *et. al.*, 2005).

Depois de terminada a fase de treinamento tem-se um classificador que deve ser capaz de predizer corretamente o rótulo de novos exemplos, que ainda não foram rotulados (REZENDE, 2005).

2.3 CONCEITOS FUNDAMENTAIS DE CLASSIFICAÇÃO HIERÁRQUICA

A grande maioria dos problemas de classificação descritos na literatura de Aprendizagem de Máquina envolve classificação plana. Nesse tipo de classificação, cada exemplo é associado a uma classe pertencente a um conjunto finito de classes, não considerando a existência de relacionamentos entre elas. No entanto, existe um grande número de problemas em que as classes são dispostas em uma estrutura hierárquica, tal como uma árvore ou DAG (do inglês, *Directed Acyclic Graph*). Esses problemas são conhecidos na literatura como problemas de classificação hierárquica (CARVALHO; FREITAS, 2007). Dessa forma, um problema de classificação hierárquica difere de um problema de classificação plana fundamentalmente na forma como as suas classes estão relacionadas entre si.

Classificação hierárquica é um tipo de problema de classificação no qual as classes envolvidas podem ser divididas em subclasses ou agrupadas em superclasses, dando origem a uma hierarquia. Nos métodos hierárquicos de classificação, o algoritmo de aprendizagem induz um modelo que captura os relacionamentos mais relevantes entre as classes funcionais no conjunto de dados de treinamento, considerando os relacionamentos hierárquicos entre elas (CARVALHO; FREITAS, 2007).

Um importante aspecto que caracteriza um problema de classificação hierárquica é o tipo de hierarquia empregado para representar os relacionamentos entre as classes. Existem duas maneiras em que as classes podem estar dispostas hierarquicamente: como uma árvore ou uma DAG.

A principal diferença entre a estrutura em árvore apresentada Figura 3 e a estrutura DAG na Figura 4 é que, na estrutura em árvore, cada nó, exceto o nó-raiz,

tem somente um nó-pai, enquanto que no DAG cada nó, exceto o nó-raiz, pode ter um ou mais nós-pai.

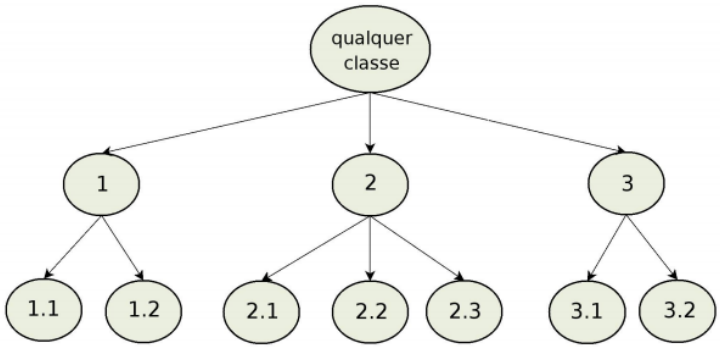

Figura 3 - Exemplo de uma hierarquia de classes estruturada como uma árvore
Fonte: Carvalho *et al.* (2007)

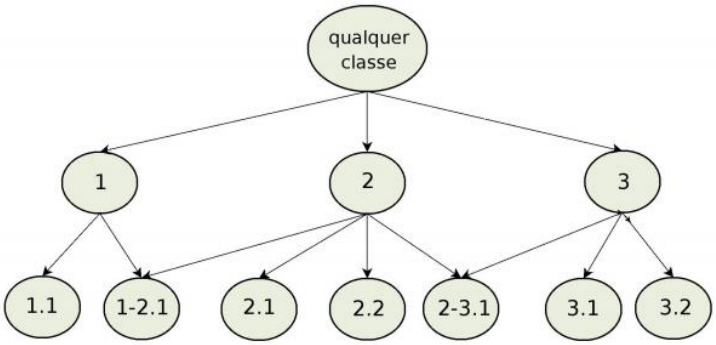

Figura 4 - Exemplo de uma hierarquia de classes estruturada como um DAG
Fonte: Carvalho *et al.* (2007)

Problemas de classificação hierárquica têm por objetivo a classificação de cada novo dado de entrada em um dos nós-folhas, pois quanto mais profunda a classe na hierarquia, mais especifico e útil é o conhecimento (CARVALHO; FREITAS, 2007). Pode ocorrer, no entanto, do classificador não apresentar uma confiabilidade desejada na classificação em uma das classes do nível mais profundo, sendo mais seguro realizar a classificação nos níveis mais elevados.

2.4 CLASSIFICAÇÃO MULTIRRÓTULO

Existe uma grande quantidade de problemas em que alguns exemplos dos dados podem pertencer a mais de uma classe (rótulo) simultaneamente. Esse grupo de problemas é conhecido como classificação multirrótulo (TROHIDIS; TSOUMAKAS; VLAHAVAS, 2008).

Segundo Trohidis *et al.* (2008), para uma descrição formal, seja $L = \lambda_j : j = 1, \dots, M$ um conjunto finito de classes (rótulos) em uma tarefa de aprendizado multirrótulo e $D = \{(\vec{x}_i | L_i), i = 1, \dots, N\}$ o conjunto de exemplos de treinamento multirrótulo, onde $\vec{x}_i$é o vetor de características (atributos) e o $L_i \subseteq Y$ conjunto de classes da i-ésima instância. No Quadro 1, são apresentados quatro exemplos rotulados com um ou mais rótulos.

Exemplo	Rótulos
1	$\{\lambda_1, \lambda_4\}$
2	$\{\lambda_3, \lambda_4\}$
3	$\{\lambda_1\}$
4	$\{\lambda_2, \lambda_3, \lambda_4\}$

Quadro 1 - Exemplo de uma base de dados multirrótulo
Fonte: Tsoumakas *et al.* (2010)

No passado, o estudo de métodos de classificação multirrótulo era motivado principalmente pelas tarefas de classificação de textos e diagnósticos médicos. Em um problema de classificação de textos, cada documento pode pertencer simultaneamente a mais de uma classe (ou tópico). Um documento, por exemplo, pode ser classificado como pertencente à área de Ciência da Computação e Física. Um artigo de jornal pode ser classificado como artes, cinema e religião.

Na área de diagnósticos médicos, um paciente pode sofrer de diabetes e câncer de próstata ao mesmo tempo (TSOUMAKAS; KATAKIS, 2007). A área de classificação de textos é a que tem maior aplicação de técnicas de classificação multirrótulo (GONÇALVES; QUARESMA, 2003; LAUSER; HOTHO, 2003; LUO; ZINCIR-HEYWOOD, 2005). Entretanto, muitos trabalhos podem ser encontrados nas áreas de Bioinformática (CLARE; KING, 2001; ZHANG; ZHOU, 2007; ELISSEEFF; WESTON, 2006), diagnóstico médico (KARALIC; PIRNAT, 1991) e classificação de imagens (BOUTELL *et al.*, 2004; SHEN *et al.*, 2004).

A Figura 5 apresenta uma comparação entre um problema de classificação convencional, em que exemplos podem ser associados a apenas uma classe, e um problema de classificação multirrótulo. Na Figura 5.a é ilustrado um problema de classificação no qual um documento pode pertencer ou à classe “Física” ou à classe “Ciência da Computação”, mas nunca às duas classes ao mesmo tempo. A Figura 5.b ilustra um exemplo de classificação multirrótulo, em que os documentos

pertencentes simultaneamente às classes “Física” e “Ciência da Computação” são classificados na classe “Física Computacional”.

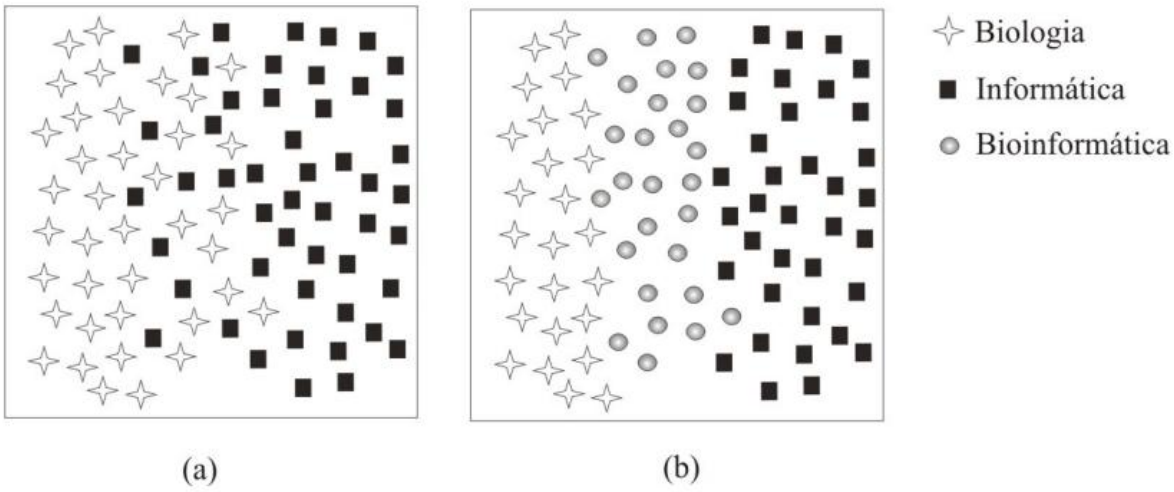

Figura 5 - (a) Típico problema de classificação (b) Problema de classificação multirrótulo
Fonte: Santos (2012)

Diferentes técnicas têm sido propostas na literatura para tratar problemas de classificação multirrótulo. Em algumas dessas técnicas, classificadores simples-rótulo podem ser combinados para tratar problemas de classificação multirrótulo. Outras técnicas modificam classificadores simples-rótulo, através de adaptações em seus mecanismos internos, para permitir que sejam utilizados em problemas multirrótulo. Ainda, novos algoritmos podem ser desenvolvidos especificamente para tratar problemas de classificação multirrótulo (CARVALHO; FREITAS, 2007).

Essas técnicas, apresentadas na Figura 6, podem ser divididas em duas abordagens principais: abordagem independente de algoritmo e abordagem dependente de algoritmo.

Na abordagem independente de algoritmo, também chamada de transformação do problema, problemas de classificação multirrótulo são tratados utilizando qualquer algoritmo de classificação tradicional. A ideia é transformar o problema original em um conjunto de problemas de classificação simples-rótulo. Essa transformação pode ser realizada baseando-se nos rótulos de classe ou nos exemplos. Na transformação baseada em rótulos de classe, conhecida como *Binary Relevance*, são utilizados L classificadores, sendo L o número de classes que estão envolvidas no problema. Para cada classificador é então associado a uma classe e treinado para resolver um problema de classificação binária, na qual é considerada a classe a qual ele está associado contra todas as outras classes envolvidas. Já na

transformação baseada nos exemplos, o conjunto de classes associado a cada exemplo é definido de maneira a converter o problema multirrótulo original em um ou mais problemas simples-rótulo (CARVALHO; FREITAS, 2007).

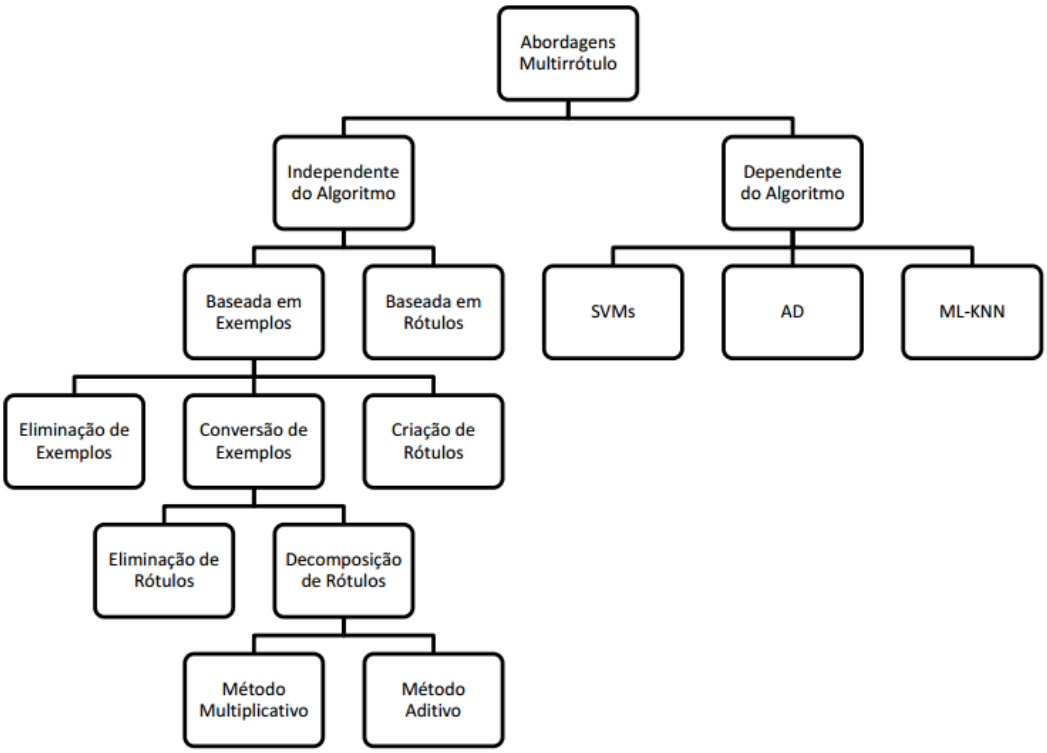

Figura 6 - Técnicas para classificação multirrótulo
Fonte: Cerri (2010)

Por sua vez na abordagem dependente de algoritmo, como o próprio nome sugere, novos algoritmos são propostos para tratar problemas de classificação multirrótulo. Tais algoritmos podem ser desenvolvidos especificamente para classificação multirrótulo ou serem baseados em técnicas de classificação convencionais, como SVMs e árvores de decisão (CARVALHO; FREITAS, 2009).

2.4.1 Métodos de Classificação Multirrótulo

Os métodos de classificação multirrrótulo podem ser divididos em dois grupos (TSOUMAKAS; KATAKIS, 2007): Transformação do problema e Adaptação de algoritmos.

2.4.1.1Transformação do problema

Como citado anteriormente, a abordagem de transformação do problema, também chamada de abordagem independente de algoritmo, consiste na conversão de uma tarefa multirrótulo em um conjunto de tarefas unirrótulo. A ideia é transformar o problema original em um conjunto de problemas de classificação unirrótulo. Assim, torna-se possível a utilização de qualquer algoritmo tradicional de classificação para tratar o problema multirrótulo (TSOUMAKAS; KATAKIS, 2007).

Na literatura há vários trabalhos propondo métodos de transformação simples que podem ser usados para converter um conjunto de dados multirrótulo em um conjunto de dados unirrótulo com o mesmo conjunto de rótulos, como por exemplo em (BOUTELL *et al.*, 2004; TSOUMAKAS; KATAKIS, 2007; KATAKIS; TSOUMAKAS; VLAHAVAS, 2008; TSOUMAKAS; KATAKIS; VLAHAVAS, 2010). Dentre os métodos utilizados podem ser citados: LP (do inglês, *Label Powerset*) (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010), BR (do inglês, *Binary Relevance*) (TSOUMAKAS; KATAKIS, 2007), RAkEL (do inglês, *RAndom k-LabELsets*) (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010), dentro outros.

2.4.1.1.1 Label powerset – LP

O método LP proposto em Tsoumakas *et al.* (2010) é um método simples de transformação do problema multirrótulo em um problema multiclasse. Ele considera cada conjunto de rótulos que existe em um conjunto de treinamento multirrótulo como uma das classes de uma nova tarefa de classificação de rótulo único (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

Com o método, os multirrótulos são transformados em unirrótulos, utilizando por exemplo a concatenação dos múltiplos rótulos para a criação de um novo unirrótulo. Outras transformações podem ser utilizadas, como a criação de um índice único para cada combinação de multirrótulos no conjunto de treinamento.

O Quadro 2 mostra o resultado da transformação do conjunto de dados do Quadro 1 usando o método *Label Powerset* e a Figura 7 ilustra o processo utilizado pelo LP.

Exemplo	Rótulos
1	$\{\lambda_1, \lambda_4\}$
2	$\{\lambda_3, \lambda_4\}$
3	$\{\lambda_1\}$
4	$\{\lambda_2, \lambda_3, \lambda_4\}$

Quadro 2 - Conjunto de dados transformados usando o método *Label Powerset*
Fonte: Tsoumakas *et al.* (2010)

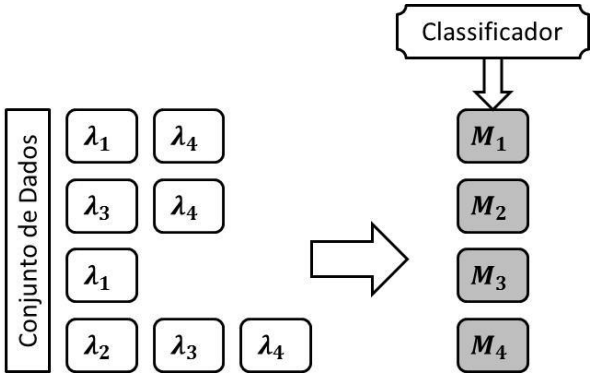

Figura 7 - Processo utilizado pelo LP
Fonte: Tsoumakas *et al.* (2010)

Uma desvantagem desse método é que a transformação do problema multirrótulo para problemas unirrótulo pode resultar em uma tarefa com muitas classes a serem preditas, uma vez que devem ser consideradas todas as combinações únicas de rótulos presentes no conjunto de treinamento como valores distintos do atributo classe. Outro problema recorrente nessa abordagem é o desbalanceamento entre as classes, onde, usualmente muitas classes apresentam frequência baixa enquanto que poucas classes apresentam alta frequência (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

A vantagem desse método, além da simplicidade, é que ele mantém a relação entre os rótulos de um mesmo exemplo, o que pode incrementar a precisão dos modelos gerados usando este método.

2.4.1.1.2 Binary relevance– BR

Dentre todos os métodos de transformação do problema, o mais popular é o *Binary Relevance* – BR (TSOUMAKAS; KATAKIS, 2007). Trata-se de um método

que realiza a decomposição do problema multirrótulo em vários subproblemas binários.

Nesse método é construído um classificador para cada classe com um mesmo algoritmo de aprendizado de máquina supervisionado. Para isso, inicialmente o conjunto de dados de treinamento, cujos exemplos de treinamento possuem mais de um rótulo é transformado em $|L|$ problemas de classificação unirrótulo binário, onde $|L|$ é a quantidade de rótulos contidos em L. Assim, pode-se afirmar que a predição de cada rótulo é considerada como uma tarefa independente.

Para cada problema unirrótulo binário, o conjunto de exemplos é replicado e os rótulos desses exemplos são modificados, de modo que cada exemplo seja positivo ou negativo para esse problema. No Quadro 3, são apresentados os conjuntos de dados produzidos ao utilizar o método *Binary Relevance* e a Figura 8 ilustra o processo utilizado pelo BR.

Exemplo	Rótulos
1	$\{\lambda_1\}$
2	$\{\neg\lambda_1\}$
3	$\{\lambda_1\}$
4	$\{\neg\lambda_1\}$

Exemplo	Rótulos
1	$\{\neg\lambda_2\}$
2	$\{\neg\lambda_2\}$
3	$\{\neg\lambda_2\}$
4	$\{\lambda_2\}$

Exemplo	Rótulos
1	$\{\neg\lambda_3\}$
2	$\{\lambda_3\}$
3	$\{\neg\lambda_3\}$
4	$\{\lambda_3\}$

Exemplo	Rótulos
1	$\{\lambda_4\}$
2	$\{\lambda_4\}$
3	$\{\neg\lambda_4\}$
4	$\{\lambda_4\}$

Quadro 3 - Conjunto de dados obtidos ao utilizar o método *Binary Relevance* nos dados do Quadro 1
Fonte: Tsoumakas *et al.* (2010)

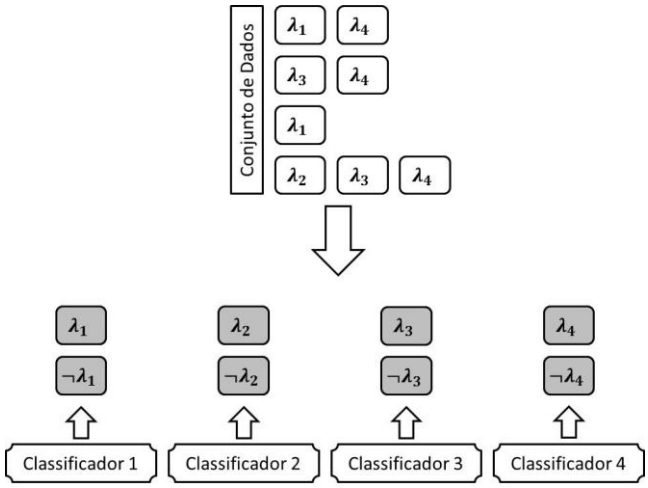

Figura 8 - Processo utilizado pelo BR
Fonte: Tsoumakas *et al.* (2010)

A principal desvantagem desse método é não considerar a dependência de rótulos na construção do modelo de classificação multirrótulo, uma vez que cada classificador binário é construído de maneira independente dos demais (TSOUMAKAS; KATAKIS, 2007).

Trata-se de um método simples e eficiente, apresentando complexidade linear com o número de monorrótulos no conjunto de dados. Além disso, a característica de independência entre os classificadores binários permite que todos os classificadores sejam construídos em paralelo, diminuindo ainda mais o tempo para construção do modelo final.

2.4.1.1.3 Random k-labelsets – RakEL

Tendo como objetivo minimizar os problemas do *Label Powerset* mencionados anteriormente, em Tsoumakas *et al.* (2010) foi proposta uma abordagem na qual, ao mesmo tempo em que são consideradas as correlações entre os rótulos, busca-se evitar o problema de suscetibilidade à ocorrência de muitas classes com poucos exemplos do LP. Nessa nova abordagem, chamada RAkEL, (do inglês, *RAndom k-labELsets*), k é um parâmetro que especifica o tamanho dos subconjuntos (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

A evidência empírica indica que RAkEL consegue melhorar substancialmente ao longo LP, especialmente em domínios com grande número de rótulos e apresenta um desempenho competitivo em relação a outras de alto desempenho métodos de aprendizagem multirrótulo (TSOUMAKAS *et al.*, 2011).

No RAkEL é construído um comitê (ensemble) de classificadores LP, onde cada classificador é treinado usando um diferente subconjunto aleatório de *labelsets*. Assim pode-se afirmar que no RAkEL, os classificadores unirrótulo, além de considerar as correlações entre os rótulos, são aplicados em subtarefas com um número gerenciável de rótulos e número adequado de exemplos por rótulo (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

O Quadro 4 mostra o resultado da transformação do conjunto de dados do Quadro 1 usando o método RAkEL, enquanto que a Figura 9 ilustra o processo realizado.

Ainda em Tsoumakas *et al.* (2010) é descrita uma extensão do RAkEL, chamada RAkELd (do inglês, *RAndom k-labELsets disjoint*). Enquanto no RAkEL original é permitida a sobreposição dos *labelsets*, no RAkELd, os *labelsets* de cada classificador do ensemble são disjuntos (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010).

Exemplo	Rótulos	Metarrótulo
1	$\{\lambda_1, \lambda_4\}$	M_1
2	$\{\lambda_3, \lambda_4\}$	M_2
3	$\{\lambda_1\}$	M_3
4	$\{\lambda_2, \lambda_3, \lambda_4\}$	M_4

Quadro 4 - Conjunto de dados transformados usando o método RAkEL
Fonte: Tsoumakas *et al.* (2010)

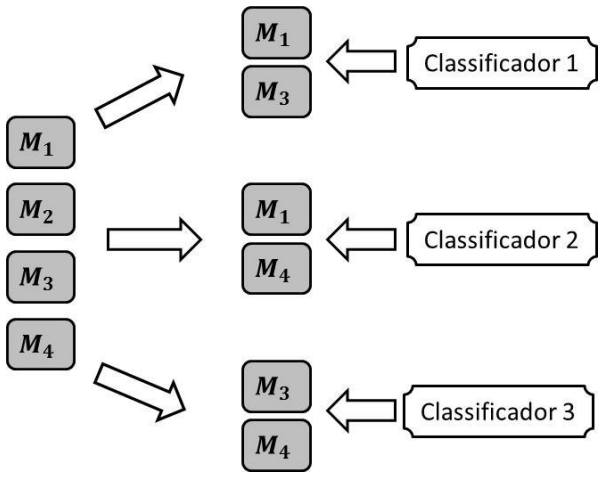

Figura 9 - Processo utilizado pelo RAkEL
Fonte: Tsoumakas *et al.* (2010)

2.4.1.2Adaptação de algoritmos

Conforme mencionado anteriormente, os métodos de adaptação de algoritmos são gerados como resultado de uma extensão de um determinado algoritmo de aprendizado unirrótulo, possibilitando assim a manipulação de dados multirrótulo diretamente.

Na literatura, há uma variedade de trabalhos tratando da adaptação das mais variadas técnicas de aprendizado de máquina, como por exemplo: Algoritmos de Indução de Árvores de Decisão (CLARE; KING, 2001), k vizinhos mais próximos (ZHANG; ZHOU, 2007), *AdaBoost* (SCHAPIRE; SINGER, 2000), dentre outras.

2.4.1.2.1 Árvore de decisão

Uma maneira natural e intuitiva de se classificar um padrão é por meio de uma sequência de decisões, em que a próxima decisão depende da decisão atual (MITCHELL, 1997). Essa sequência de decisões pode ser representada por uma estrutura de dados do tipo árvore, a qual é definida recursivamente como: um nó folha que corresponde a uma classe ou um nó interno de decisão, que contém uma decisão sobre algum atributo. Para cada resultado da decisão, existe uma aresta para uma subárvore. Essa técnica de classificação está entre as mais populares e tem sido aplicada em tarefas como, por exemplo, diagnóstico médico (MITCHELL, 1997).

Na Figura 10 é apresentado um exemplo ilustrativo de uma árvore de decisão que utiliza informações climáticas para realizar a inferência se será possível jogar tênis em um determinado dia.

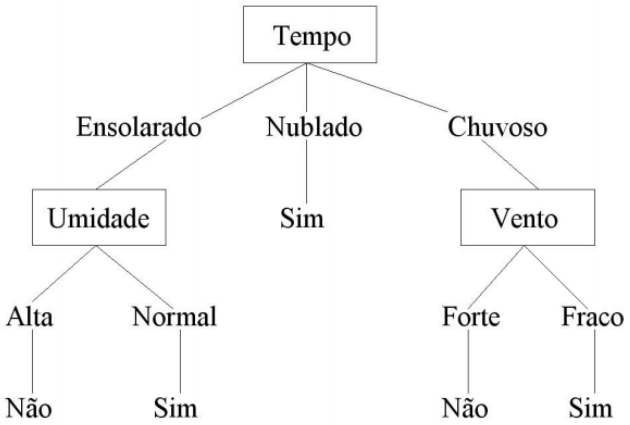

Figura 10 - Exemplo de árvore de decisão
Fonte: Mitchell (1997)

Clare *et al.*(2001), em seu trabalho adaptou a fórmula de cálculo da entropia de modo a viabilizar a manipulação de dados multirrótulo. Em um dos algoritmos mais utilizados para construção de árvores de decisão, chamado J48, os nós da árvore de decisão são definidos através dessa medida de entropia (CLARE; KING,

2001). A fórmula dessa medida, originalmente elaborada para problemas unirrótulo, foi modificada, de modo a permitir seu uso em problemas multirrótulo. A fórmula de cálculo da entropia foi modificada como segue na Equação 2.1.

$$Entropia(D) = -\Sigma P(\lambda_j) * logP(\lambda_j) + q(\lambda_j) * logq(\lambda_j) \quad (2.1)$$

em que :

D é o conjunto de exemplos multirrótulo;

$P(\lambda_j)$é a frequência relativa da classe λ_j;

$q(\lambda_j)$é$1 - P(\lambda_j)$.

Outra modificação proposta por Clare *et al.* (2001) altera a funcionalidade dos nós-folha da árvore de decisão, os quais passam a representar um subconjunto de rótulos e não apenas um único rótulo como no algoritmo original (CLARE; KING, 2001).

2.4.1.2.2 Multi-Label k-Nearest Neighbor

No trabalho de Zhang *et al.* (2007) foi proposto o método *MultiLabel k-Nearest Neighbors* (ML-kNN), diferenciado pelo uso de probabilidades a *priori* e a *posteriori*.

O ML-kNN é um classificador multi-label construído baseado no popular método kNN (FIX; HODGES, 1951). Para cada d_j elemento de teste, o ML-kNN inicialmente encontra seus k vizinhos mais próximos no conjunto de treino usado, isto é, encontra os k primeiros elementos ordenados pelo valor de similaridade com d_j de forma decrescente, usando por exemplo, a distância Euclidiana. Mais tarde, o algoritmo identifica quantos exemplares de cada categoria existem dentre os k vizinhos mais próximos de d_j, que chamaremos de k_i ($i \in \{1, 2, \dots, |C|\}$), onde $|C|$ é o número de categorias. Seja H_1^i o evento em que d_j possui o rótulo i e H_0^i o evento em que d_j não possui o rótulo i. E mais, seja E_j^i o evento em que existem j vizinhos mais próximos de d_j pertencentes á categoria i. Assim temos:

$$y_{d_j}(i) = argmax_{b \in 0,1} P(H_b^i) P(E_j^i | H_b^i) \quad (2.2)$$

Na Equação 2.2, $y_{d_j}(i)$ é a probabilidade da amostra d_j pertencer à categoria i. $P(H_b^i)$, onde ($i \in \{1,2,\dots,|C|\}$ e $b \in \{0,1\}$) e $P(E_j^i|H_b^i)$ ($j \in \{0,1,\dots,k\}$), são respectivamente, a probabilidade *priori* e *posteriori* da categoria i. Essas probabilidades são calculadas na etapa de treino. Primeiramente é estimada a probabilidade *priori* de cada categoria usando a equação 2.3.

$$P(H_1^i) = \frac{\delta + N_i}{2\delta + N} \qquad P(H_0^i) = 1 - P(H_1^i) \tag{2.3}$$

$N_i$denota o número de exemplos da categoria i no conjunto de treinamento e N denota o número total de exemplos. δ é um parâmetro para suavizar a probabilidade.

Após isso, o ML-kNN faz o seguinte: para cada exemplas w_y no conjunto de treinamento, onde ($y \in \{1,2,\dots,N\}$), o algoritmo encontra seus k vizinhos mais próximos e calcula o número total de votos que cada categoria recebe dos k vizinhos mais próximos. Em outras palavras, seja k_i o número de votos que cada categoria i recebeu de w_y, se o exemplar w_y pertence à categoria i, então será adicionado 1 a $L_{k_i}^i$, senão será adicionado 1 a $\overline{L_{k_i}^i}$. $L_{k_i}^i$e$\overline{L_{k_i}^i}$ indicam quantos exemplares de treino estão relacionados com a categoria i, e respectivamente, não relacionados com a categoria i, e respectivamente, não relacionados com a categoria i. Finalmente com essas informações, as probabilidades *posteriori* são calculadas como descrito nas Equações 2.4 e 2.5:

$$P(E_j^i|H_1^i) = \frac{\delta + L_j^i}{\delta(k+1) + \sum_{o=0}^{k} L_o^i} \tag{2.4}$$

$$P(E_j^i|H_0^i) = \frac{\delta + \overline{L_j^i}}{\delta(k+1) + \sum_{o=0}^{k} \overline{L_o^i}} \tag{2.5}$$

O ML-kNN precisa apenas de dois parâmetros: o número k de vizinhos mais próximos e a suavização δ da probabilidade. Nas Equações 2.3, 2.4 e 2.5 o valor de δ modifica levemente as probabilidades *priori* e as probabilidades *posteriori*.

Uma generalização do método ML-kNN possibilita considerar a dependência de rótulos durante o aprendizado multirrótulo (YOUNES *et al.*, 2011). De modo similar a outros algoritmos baseados no kNN, cada exemplo a ser predito nesse método tem seus k vizinhos identificados no conjunto de treinamento. O princípio *maximum* a *posteriori* (MAP) é utilizado em escopo global para atribuir um conjunto de rótulos para um exemplo a ser predito, de modo a oferecer suporte para o tratamento da dependência de rótulos. Esse princípio possibilita, por exemplo, que o número de rótulos distintos na vizinhança seja considerado durante o processo de predição, diferentemente do que ocorre no ML-kNN (ZHANG; ZHOU, 2007).

2.4.1.2.3 AdaBoost

O *AdaBoost* é um algoritmo de aprendizado supervisionado do tipo *boost*. O *AdaBoost* combina um conjunto de funções simples de classificação, denominadas classificadores fracos para formar um classificador forte (FREUND; SCHAPIRE, 1995).

Segundo Freund *et al.* (1995), um classificador forte é composto de um conjunto de classificadores fracos, associados a pesos que classificam de forma precisa dois conjuntos de dados pré-rotulados, onde as características com pesos maiores são mais significativas para a classificação de exemplos definidos como parte de um certo conjunto (FREUND; SCHAPIRE, 1995).

Em Schapire *et al.* (1999) e Schapire *et al.*(2000), são propostas duas extensões para o algoritmo *AdaBoost* (FREUND; SCHAPIRE, 1995), de maneira a permitir seu uso em problemas multirrótulos. Uma das adaptações, denominada *AdaBoost.MH*, foi feita uma modificação na maneira de se avaliar o desempenho preditivo do modelo induzido, verificando sua capacidade de predizer um conjunto correto de classes para um dado exemplo. A segunda adaptação, denominada *AdaBoost.MR*, foi feita uma mudança no algoritmo que faz com que ele passe a predizer um *ranking* de classes para cada exemplo de entrada.

2.4.1.2.4 Métodos probabilísticos

Esse método assume a probabilidade de um evento A (ex. um paciente doente) dado um evento B ocorrer (ex. o resultado de exames realizados nesse paciente é positivo). No contexto de AM (MITCHELL, 1997), a probabilidade é a frequência relativa de objetos que contém o valor de uma classe no conjunto de dados. Os objetos são descritos por um conjunto de valores de atributos de entrada que representa o evento B.

A estimativa da probabilidade $P(A|B)$ de que um evento B ocorra para cada classe (ou evento A) pode ser calcula por meio do Teorema de Bayes dada na Equação 2.6.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.6}$$

em que:

$P(A)$é a probabilidade a priori da classe;

$P(B|A)$é a probabilidade de observar vários objetos que pertencem à classe;

$P(B)$é a probabilidade de ocorrência desses objetos.

Em Mccallum (1999), é proposto um modelo probabilístico generativo, aplicado à tarefa de classificação de documentos. Neste modelo, é utilizada uma abordagem de classificação bayesiana, na qual cada rótulo gera diferentes palavras. Baseado neste modelo, um documento multirrótulo é produzido por uma mistura das distribuições das palavras de seus rótulos (MCCALLUM, 1999).

Em seu trabalho, Ghanrawi *et al.* (2005), explora o uso de campos aleatórios condicionais no qual dois modelos gráficos que parametrizam co-ocorrências de rótulos são apresentados. O primeiro, multirrótulo coletivo, captura padrões de co-ocorrências entre rótulos, enquanto que o segundo, multirrótulo coletivo com características, tenta capturar o impacto que uma característica individual tem sobre a probabilidade de co-ocorrência de um par de rótulos (GHAMRAWI; MCCALLUM, 2005).

2.5 CLASSIFICAÇÃO HIERÁRQUICA MULTIRRÓTULO

A classificação hierárquica multirrótulo tem surgido como uma nova categoria de problemas de classificação, com características tanto dos problemas de classificação multirrótulo, quanto de problemas de classificação hierárquica. Problemas pertencentes a esta nova categoria são denominados de problemas de classificação hierárquica multirrótulo (HMC, do inglês *Hierarchical Multilabel Classification*).

Em um problema de classificação hierárquica multirrótulo, um exemplo pode pertencer a múltiplas classes ao mesmo tempo e essas classes são organizadas de maneira hierárquica. Dessa forma, um exemplo pertencente a uma classe, automaticamente pertence a todas as suas superclasses (CERRI; CARVALHO, 2010).

Segundo Blockeel *et al.* (2006), dado um espaço de exemplos X e uma hierarquia de classes $(C, \leq_h)$, no qual C é um conjunto de classes e $\leq_h$ é uma ordem parcial estruturada como uma árvore enraizada, representando o relacionamento de superclasse (para todo $c_1, c_2 \in C : c_1 \leq_h c_2$ se e somente se c_1 é uma superclasse de c_2); um conjunto T de exemplos (x_i, Y_i) com $x_i \in X$ e $Y_i \subseteq C$ tal que $c \in Y_i \Rightarrow \forall c' \leq_h c : c' \in Y_i$; e algum critério de qualidade q (que tipicamente recompensa modelos com alta acurácia preditiva e baixa complexidade. Encontrar uma função $f : X \rightarrow 2^C$ (na qual 2^C é o conjunto potência de C), tal que $c \in f(x) \Rightarrow \forall c' \leq_h c : c' \in f(x)$ e f maximiza q (BLOCKEEL, 2006).

Na Figura 11, é apresentado um exemplo de problema de classificação hierárquica multirrótulo, onde a hierarquia de classes é representada através de uma árvore. Na Figura 12, é apresentado um exemplo de predição hierárquica multirrótulo onde uma notícia de um jornal está associado à Ciência da Computação e Futebol, e pode então ser classificado tanto como Ciências/Computação, quanto como Esportes/Coletivos/Futebol. Como saída, do processo de predição é obtida uma subárvore da árvore original.

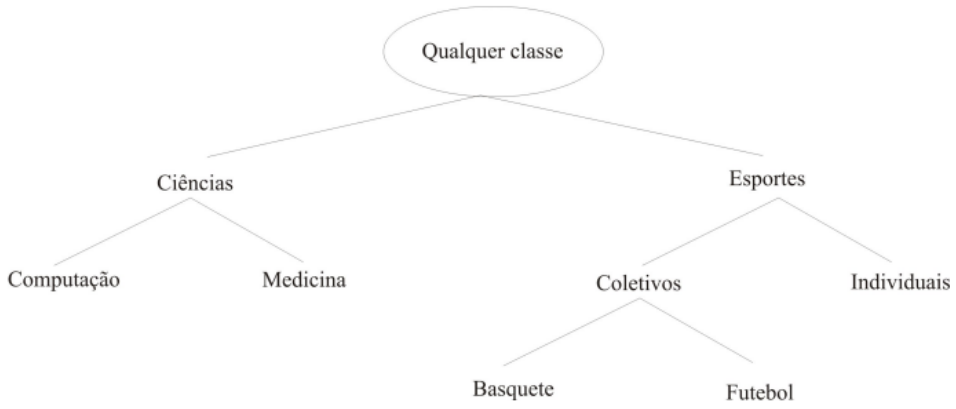

Figura 11 - Hierarquia de classes de um problema hierárquico multirrótulo estruturado como uma árovre
Fonte: Cerri *et al.* (2010)

O critério de qualidade q pode ser baseado na distância entre as classes da hierarquia, podendo ser, por exemplo, a precisão média das diferentes classes preditas, ou pode considerar que erros de classificação em níveis da hierarquia mais próximos da raiz são piores que em níveis mais profundos (STRUYF; BLOCKEEL; CLARE, 2005).

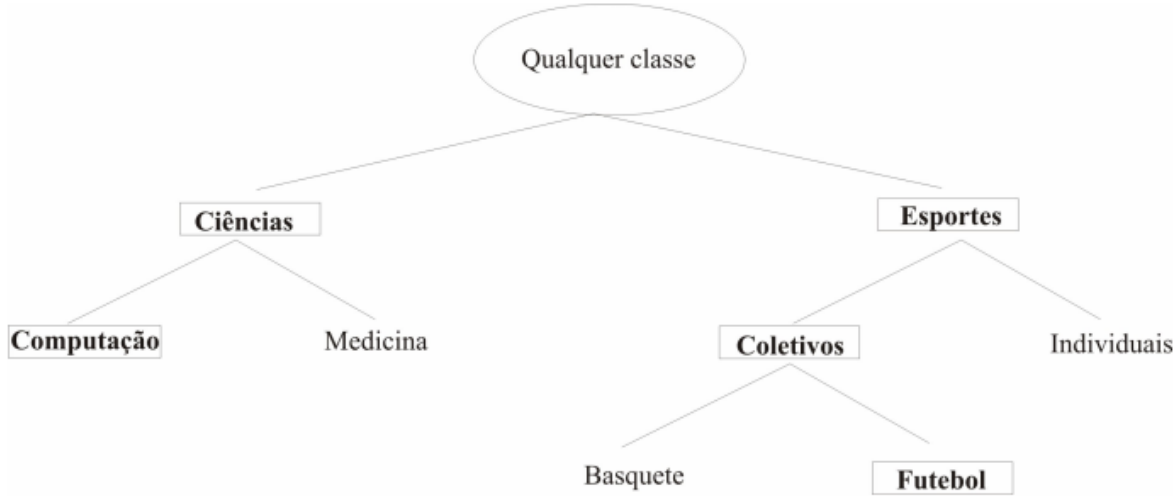

Figura 12 - Predições gerando uma subárvore
Fonte: Cerri *et al.* (2010)

Esse tipo de problema é bastante comum, principalmente em problemas de categorização de texto e classificação de genes e proteínas quanto a suas funções. Pode-se dizer também que problemas de classificação hierárquica multirrótulo são mais complexos que os demais problemas de classificação, uma vez que as classes envolvidas no problema, além de estarem estruturadas em uma hierarquia, os

exemplos podem pertencer a mais de uma classe ao mesmo tempo (CERRI; CARVALHO, 2010).

2.5.1 Métodos de Classificação Hierárquica Multirrótulo

Vários métodos podem ser utilizados no tratamento de tarefas de classificação hierárquica multirrótulo. Na literatura, há vários trabalhos propondo e analisando abordagens e métodos para tratamentos de problemas hierárquico multirrótulo(HMC) de diferentes domínios, tais como: classificação de texto (ROUSU *et al.*, 2006), predição de funções genômicas e proteínas (BLOCKEEL *et al.*, 2006; CLARE; KING, 2003; STRUYF, *et al.*, 2005; BARUTCUOGLU *et al.*, 2008; VENS *et al.*, 2008). Contudo, não há um consenso sobre que abordagem utilizar para o tratamento de problemas hierárquicos multirrótulo.

O Quadro 5 apresenta os trabalhos organizados de acordo com a abordagem utilizada para a resolução do problema hierárquico multirrótulo, e o Quadro 6 apresenta, de forma resumida, alguns desses trabalhos.

Artigo	**Abordagem Utilizada**
(Clare; King, 2001)	Predição hierárquica com algoritmos de classificação não hierárquicos
(Blockeel *et al.*, 2006)	Abordagem *One-Shot*
(Clare; King, 2003)	Abordagem *One-Shot*
(Kiritchenko *et al.*, 2004)	Abordagem *Top-Down*
(Struyf *et al.*, 2005)	Abordagem *One-Shot*
(Barutcuoglu *et al.*, 2006)	Utiliza classificadores binários para cada classe. Suas predições são combinadas com aprendizado Bayesiano
(Rousu *et al.*, 2006)	Abordagem *One-Shot*
(Esuli *et al.*, 2006)	Abordagem *Top-Down*
(Blockeel *et al.*, 2006)	Abordagem*Top-Down* e *One-Shot*
(Cesa-Bianchi *et al.*, 2006)	Abordagem *One-Shot*
(Alves *et al.*, 2010)	Abordagem *Top-Down*
(Vens *et al.*, 2008)	Predição hierárquica com algoritmos de classificação não hierárquica, Abordagem *Top-Down* e *One-Shot*
(Valentini, 2009)	Combinação de classificadores

Quadro 5 - Abordagens para tratar problemas hierárquicos mutirrótulo
Fonte: Cerri *et al.* (2010)

Artigo	Descrição
(Clare; King, 2001)	Variação do algoritmo C4.5, utilizando a soma da entropia de todas as classes. Utiliza classificadores individuais para cada nível da hierarquia.
(Blockeel *et al.*, 2006)	Utiliza um algoritmo baseado na noção de predictiveclusteringtrees, onde árvores de decisão são vistas como uma hierarquia de clusters. Gera uma única árvore de decisão para a hierarquia.
(Clare; King, 2003)	Extensão de Clare*et al.* (2001), predizendo classes em todos os níveis da hierarquia, atribuindo um custo de classificação errada maior para níveis mais altos. Utiliza um único classificador.
(Kiritchenko *et al.*, 2004)	Utiliza uma técnica baseada na técnica *Top-Down*, utilizando técnicas de AM que têm como saída uma pontuação de probabilidade.
(Struyf *et al.*, 2005)	Utiliza o mesmo conceito de *predictiveclusteringtrees* utilizado por Blockeel*et al.* (2002), mas foi introduzida uma nova medida de distância especifica para classificação hierárquica multirrótulo.
(Barutcuoglu *et al.*, 2006)	Utiliza uma hierarquia de classificadores SVMs. Os classificadores são treinados e então suas predições são combinadas utilizando aprendizado Bayesiano.
(Rousu *et al.*, 2006)	Propõe um algoritmo baseado em *kernel* para a classificação hierárquica multirrótulo. A hierarquia de classificação é representada como uma árvore Markov. É treinado um único classificador para toda a hierarquia.
(Esuli *et al.*, 2006)	Utiliza um algoritmo chamado *TreeBoost.MH*, que consiste de uma variação hierárquica do algoritmo *AdaBoost.MH*, membro da família de algoritmos *boosting*.
(Blockeel *et al.*, 2006)	Compara métodos baseados em árvore de decisão. Compara um método que induz uma única árvore de decisão para todas as classes com um método que induz uma árvore de decisão para cada classe.
(Cesa-Bianchi *et al.*, 2006)	Treina um classificador incrementalmente para cada nó da hierarquia de classes, baseado no método *Top-Down*.
(Alves *et al.*, 2010)	Utiliza um sistema imunológico inteligente para a tarefa de classificação. O algoritmo tem como objetivo encontrar um conjunto de regras que sejam tanto hierárquicas quanto multirrótulo.
(Vens *et al.*, 2008)	Compara abordagens baseadas na noção de *predictiveclusteringtrees*. Compara três abordagens de indução de árvores de decisão. A primeira faz predições para todas as classes da hierarquia de uma única vez. A segunda define uma tarefa de classificação simples-rótulo independente para cada classe, ignorando relacionamentos hierárquicos. A terceira induz uma árvore de decisão para cada classe, porém considera os relacionamentos hierárquicos.
(Valentini, 2009)	Utiliza combinação de classificadores. As saídas de classificadores base, especializados em uma única classe, são combinadas para obter uma classificação final.

Quadro 6 - Trabalhos que tratam problemas hierárquicos multirrótulo
Fonte: Cerri *et al.* (2010)

2.5.2 Medidas de Avaliação

Apesar da maioria dos trabalhos apresentados na literatura avaliarem, de alguma maneira, o desempenho da classificação hierárquica multirrótulo, parece não haver ainda medidas definidas, ou adotadas por um grande número de trabalhos, para essa avaliação.

As medidas de avaliação podem ser divididas em algumas abordagens, sendo elas: medidas baseadas nas relações de ancestralidade e descendência, medidas baseadas em distância, medidas baseadas em similaridades. A escolha por essas medidas está em avaliar o resultado da classificação de diferentes maneiras.

2.5.2.1 Medidas baseadas nas relações de ancestralidade e descendência

Medidas baseadas em ancestralidade e descendência consideram os ancestrais e os descendentes das classes preditas no momento dos cálculos da avaliação. Dentre as medidas baseadas nesse conceito tem-se: Precisão e Revocação Hierárquica e *Hierarchical Loss Function*.

2.5.2.1.1 Precisão e revocação hierárquica

No trabalho de Kiritchenko *et al.* (2004), foram propostas duas medidas de avaliação baseadas nas medidas convencionais de precisão e revocação, levando em consideração os relacionamentos hierárquicos entre as classes. Essas medidas foram chamadas de precisão e revocação hierárquicas e levam em consideração classificações nos nós internos e nós-folha.

Cada exemplo pertence não apenas à sua classe, mas também a todos os ancestrais dessa classe na estrutura hierárquica. Dessa maneira, dado um exemplo qualquer (x_i, Y_i), com x pertencente ao conjunto X de exemplos, Y_i o conjunto de classes preditas para o exemplo x_i, e Y_i' o conjunto de classes verdadeiras do exemplo x_i, os conjuntos Y_i e Y_i' podem ser entendidos para conterem suas correspondentes classes ancestrais da seguinte maneira: $\hat{Y} = \bigcup_{y_i \in Y_i} Ancestrais(y_k)$ e $\hat{Y} = \bigcup_{y_i \in Y_i'} Ancestrais(y_i)$.

A precisão e revocação hierárquica (Prec e Rev) são calculadas utilizando as Equações 2.7 e 2.8, respectivamente.

$$Prec = \frac{\sum_i |\hat{Y}_i \cap \hat{Y}_i'|}{\sum_i |\hat{Y}_i|} \tag{2.7}$$

$$Rev = \frac{\sum_i |\hat{Y}_i \cap \hat{Y}_i'|}{\sum_i |\hat{Y}_i'|} \tag{2.8}$$

Essas medidas contam o número de classes preditas corretamente, juntamente com o número de classes ancestrais dessas classes preditas corretamente, assumindo que exemplos também pertencem aos ancestrais de suas classes corretas (KIRITCHENKO; MATWIN; FAMILI, 2004).

A precisão e a revocação hierárquicas utilizadas sozinhas não são suficientes para a avaliação de classificadores. Sendo assim, as medidas $Prec$ e Rev devem ser combinadas em uma extensão hierárquica da medida *F-Measure*, chamada FM, apresentada na Equação 2.9. Na Equaçãoβ, refere-se à importância atribuída aos valores de $Prec$ e Rev. Quando aumenta-se o valor de β, aumenta-se o peso atribuído ao valor de Rev, e quando diminui-se β, aumenta-se o peso atribuído ao valor de $Prec$.

$$FM = \frac{(\beta^2 + 1) \times Prec \times Rev}{\beta^2 \times Prec \times Rev} \tag{2.9}$$

2.5.2.1.2 Hierarchical Loss Function

No trabalho de Cesa-Bianchi *et al.* (2006), foi proposta uma nova medida de avaliação chamada de *Hierarchical Loss Function* (H-Loss). Essa medida utiliza a noção intuitiva de que sempre que um erro de classificação é cometido em um nó da hierarquia de classes, não devem haver penalizações adicionais para erros cometidos na subárvore desse nó (CESA-BIANCHI; GENTILE; ZANIBONI, 2006).

Segundo os autores, dada uma estrutura hierárquica G, essa estrutura pode ser considerada uma floresta, composta por árvores definidas sobre o conjunto de

classes do problema. Uma classificação multirrótulo $v \in \{0,1\}^L$ respeita essa estrutura G se e somente se v for a união de um ou mais caminhos de G, onde cada caminho começa em uma raiz e necessariamente termina em uma folha. Assim, todos os caminhos em G, de uma raiz até a folha, são examinados. Sempre que um nó i é encontrado, tal que $\hat{y}_i \neq v_i$, o valor 1 é adicionado à função *H-Loss*, e todas as predições na subárvore enraizada no nó i são descartadas. Dada essa definição, pode-se dizer que $l_{0/1} \leq l_H \leq l_\Delta$. A função *H-Loss* que leva em consideração os relacionamentos hierárquicos é definida na Equação 2.10, na qual $ANC(i)$ denota o conjunto de ancestrais do nó i.

$$l_H(\hat{y}, v) = \sum_{i=1}^{L} \{\hat{y}_i \neq v_i \ \wedge \ \hat{y}_j = v_j, j \in ANC(i)\} \qquad (2.10)$$

2.5.2.2Medidas baseadas em distância

Como classes que estão mais perto umas das outras na hierarquia tendem a ser mais próximas entre si do que de outras classes, esse método considera a distância entre a classe verdadeira e a classe predita na hora de medir o desempenho do classificador.

No trabalho de Sun*et al.* (2001), foram utilizadas medidas chamadas de Micro/Macro Precisão Hierárquica e Micro/Macro Revocação Hierárquica. Segundo os autores, as medidas Micro Precisão e Revocação atribuem igual importância a todos os exemplos, enquanto as medidas Macro Precisão e revocação atribuem igual importância a todas as classes (SUN *et al.*, 2001).

As medidas macro precisão/ revocação hierárquicas primeiramente medem o desempenho obtido em cada classe do problema separadamente, e então obtém uma média desses desempenhos. As medidas micro precisão/revocação hierárquicas, por outro lado, medem o desempenho médio obtido em cada exemplo do conjunto de dados. Assim, as macro medidas são consideradas como a média do desempenho por classe, enquanto as micro medidas são consideradas a média do desempenho por exemplo do conjunto de dados (YANG, 1999).

O cálculo das medidas é realizado computando-se, para cada classe, a contribuição dos exemplos erroneamente atribuídos àquela classe. Para o cálculo

dessa contribuição, é necessário que se defina uma distância aceitável entre duas classes, dada por Dis_0, que deve ser maior que 0. A distância Dis_0 é igual ao número de arestas que separam uma classe predita de uma classe verdadeira na estrutura hierárquica.

A Contribuição de um exemplo x_j a uma classe y_i é formalmente definida pelas Equações 2.11 e 2.12, nas quais $x_j.agd$ e $x_j.lbd$ são respectivamente as classes preditas e verdadeiras do exemplo x_j.

Se x_j é um Falso Positivo:

$$Con(x_j, y_i) = \sum_{y' \in x_j.lbd}(1.0 - \frac{Dis(y', y_i)}{Dis_0}) \tag{2.11}$$

Se x_j é um Falso Negativo:

$$Con(x_j, y_i) = \sum_{y' \in x_j.lbd}(1.0 - \frac{Dis(y', y_i)}{Dis_0}) \tag{2.12}$$

Após seu cálculo, a contribuição de um exemplo x_j é então restringida à faixa de valores [-1, 1]. Esse refinamento, denotado por $RCon(x_j, y_j)$, é definido na Equação 2.13.

$$RCon(x_j, y_j) = \min(1, \max(-1, Con(x_j, y_j))) \tag{2.13}$$

Para todos os exemplos, a contribuição total de falsos positivos ($FpCon_i$) e falsos negativos ($FnCon_i$)é definida nas Equações 2.14 e 2.15.

$$FpCon_i = \sum_{x_j \in FP_i} RCon(x_j, y_j) \tag{2.14}$$

$$FpCon_i = \sum_{x_j \in FN_i} RCon(x_j, y_j) \tag{2.15}$$

Após os cálculos das contribuições de cada exemplo, são calculados os valores da Precisão e da Revocação Hierárquica de cada classe. Os cálculos são apresentados nas Equações 2.16 e 2.17.

$$Pr_i^{CD} = \frac{\max(0, |TP_i| + FpCon_i + FnCon_i}{|TP_i| + |FP_i| + FnCon_i} \tag{2.16}$$

$$Re_i^{CD} = \frac{\max(0, |TP_i| + FpCon_i + FnCon_i}{|TP_i| + |FN_i| + FpCon_i} \tag{2.17}$$

Os valores estendidos da Precisão e Revocação Hierárquicas (Micro Precisão e Revocação Hierárquicas) são apresentados nas Equações 2.18 e 1.19, em que m representa o número de classes do problema.

$$\hat{P}r^{\mu CD} = \frac{\sum_{i=1}^{m}(\max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}|TP_i| + |FP_i| + FnCon_i} \tag{2.18}$$

$$\hat{R}e^{\mu CD} = \frac{\sum_{i=1}^{m}(\max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}|TP_i| + |FN_i| + FpCon_i} \tag{2.19}$$

De acordo com o valor escolhido para Dis_0, as contribuições $FpCon_i$ e $FnCon_i$ podem ser negativas. Portanto, uma função max é aplicada ao numerador das Equações 2.18 e 2.19 para fazer com que seus valores não sejam menores que 0. Como $FpCon_i \leq |FP_i|$, quando $|TP_i| + |FP_i| + FnCon_i \leq 0$, o numerador $\max(0, |TP_i| + FpCon_i + FnCon_i) = 0$, e $\hat{P}r^{\mu CD}$ pode ser considerado 0 nesse caso. A mesma regra é aplicada ao cálculo de $\hat{R}e^{\mu CD}$ (SUN *et al.*, 2001).

Além dos cálculos da Micro Precisão e Revocação Hierárquicas, pode-se ainda obter os valores estendidos da Macro Precisão e Revocação Hierárquicas, por meio das Equações 2.20 e 2.21. Nas equações, m refere-se ao número de classes envolvidas.

$$\hat{P}r^{\mu CD} = \frac{\sum_{i=1}^{m} Pr_i^{CD}}{m} \tag{2.20}$$

$$\hat{R}e^{\mu CD} = \frac{\sum_{i=1}^{m} Re_i^{CD}}{m} \quad (2.21)$$

Assim como as medidas Precisão e Revocação Hierárquicas, utilizadas por Kiritchenko *et al.* (2004), as medidas Micro/Macro Precisão e Revocação Hierárquicas também podem ser combinadas na medida FM, calculada por meio da Equação 2.9.

Medidas baseadas em distância possuem a desvantagem de não considerarem o fato de que classificações nos níveis mais profundos da hierarquia são mais difíceis e levam a informações mais especificas do que classificações nos níveis mais altos. É possível contornar essa desvantagem, tornando mais altos os custos de classificações erradas em níveis mais elevados do que em níveis mais profundos da hierarquia.

2.5.3 Testes Estatísticos para Validação dos Resultados

A análise estatística dos resultados obtidos de um determinado estudo é uma ferramenta muito importante na validação desses dados. Os testes estatísticos são fundamentalmente utilizados em pesquisas que tem como objetivo comparar condições experimentais.

Existe uma série de testes estatísticos que podem auxiliar as pesquisas e esses testes podem ser divididos em paramétricos e não-paramétricos. A diferença entre ambos os testes refere-se ao tipo de valores da variável estudada. Nos testes paramétricos os valores da variável devem ter distribuição normal ou aproximação normal. Já os testes não-paramétricos não têm exigências quanto ao conhecimento da distribuição da variável na população.

Devido ao não conhecimento da distribuição dos dados, experimentos realizados para tratar de problemas hierárquicos multirrótulo são utilizados testes não-paramétricos, tal qual o teste de Wilcoxon (*Wilcoxon Signed Rank Test*) e o teste de Friedman.

2.5.3.1Teste de Wilcoxon

O teste de Wilcoxon (1945) é aplicado quando estão em comparação dois grupos relacionados e a variável deve ser mensuração ordinal. O teste classifica em postos a diferença entre os algoritmos sobre cada base usada para a avaliação de desempenho. Em seguida, soma as diferenças positivas apresentada na Equação 2.22 e as negativas na equação 2.23.

$$W^+ = \sum_{d_i>0} r_i \tag{2.22}$$

$$W^- = \sum_{d_i<0} r_i \tag{2.23}$$

em que r_i é o posto (*ranking*) da i-ésima base avaliada considerando as diferenças entre os algoritmos comparados.

Na sequência calcula-se o valor T, na equação 2.24, que representa a menor das somas de postos de mesmo sinal (DESMAR, 2006).

$$T = \min(W^+; W^-) \tag{2.24}$$

Em seguida determina-se o valor de N que é o total das diferenças com sinal. Se $N \leq 25$, os valores críticos de T são tabelados, onde N representa o número de base de dados avaliadas, descontados o número de empates $(d_i = 0)$. Já para os valores em que $N > 25$utiliza-se a estatística z, na Equação 2.25, pois se considera que a distribuição dos dados é aproximadamente normal.

$$z = \frac{T - \frac{1}{4}n(n+1)}{\sqrt{\frac{1}{24}n(n+1)(2n+1)}} \tag{2.25}$$

A hipótese nula assume que a diferença de desempenho entre algoritmos não é significativa. Com nível de confiança $\alpha = 0{,}05$, a hipótese nula não pode ser rejeitada se $-1{,}96 \leq z \leq 1{,}96$.

2.5.3.2Teste de Friedman

O teste de Friedman é um teste não paramétrico (FRIEDMAN, 1940) recomendado para comparar o desempenho de vários algoritmos sobre diferentes bases de dados (DESMAR, 2006).

Os algoritmos são organizados por postos (*ranking*), de acordo com o desempenho obtido em cima de cada base de dados, sendo atribuído 1 ao primeiro colocado, 2 ao segundo, e assim sucessivamente. Na sequência, calcula-se a média dos postos obtidos pelos algoritmos sobre todos os conjuntos de dados usados nos experimentos.

Após o calculo da média dos postos médios dos classificadores é necessário calcular a diferença estatística existente entre eles. Este calculo é feito através da aplicação da Equação 2.26.

$$X_F^2 = \frac{12n}{k(k+1)}\left[\sum_j r_j^2 - \frac{k(k+1)^2}{4}\right] \tag{2.26}$$

em que n é a quantidade de bases de dados, k é a quantidade de algoritmos e r_j é o posto médio para o j-ésimo algoritmo.

Segundo Iman *et al.* (1980) o teste de Friedman é considerado muito conservador, e por isso, sugere a utilização da estatística F_F de Snedcor com $k-1$ e $(k-1)*(n-1)$ graus de liberdade.

Se a hipótese nula é rejeitada Desmar (2006) sugere o teste de Nemenyi, que calcula a distância crítica (CD – *critical distance*) entre desempenhos através da Equação 2.27.

$$CD = q_a\sqrt{\frac{k(k+1)}{6N}} \tag{2.27}$$

em que os valores de q_a são tabelados conforme o nível de significância. Logo, um algoritmo é considerado estatisticamente superior a outro se a diferença entre os postos médio de cada um for maior que a distância critica calculada.

2.6 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo teve início com a apresentação dos principais conceitos básicos associados à aprendizagem de máquina e as principais características de cada uma das categorias de aprendizagem.

Foi apresentada também uma breve introdução nos conceitos fundamentais de classificação e classificação de dados hierárquica. Além dos conceitos referentes a classificação de dados multirrótulo, abordando as principais técnicas utilizadas para a resolução desse tipo de problema de classificação.

Por fim, foram apresentados os conceitos fundamentais de classificação hierárquica multirrótulo, tema do presente trabalho. Esse problema pode ser tratado como uma combinação dos problemas de classificação hierárquica com os problemas de classificação multirrótulo. Foram apresentadas as técnicas utilizadas para a resolução desse tipo de problema de classificação, bem como algumas medidas especificas utilizadas para a avaliação de classificadores hierárquicos multirrótulo. Foi feita também uma breve revisão bibliográfica dos principais trabalhos da literatura.

3 METODOLOGIA

Neste capítulo está descrita a configuração utilizada para a realização deste estudo e está organizado da seguinte forma: Na seção 3.1 serão apresentadas as bases de dados utilizadas nos experimentos. Na seção 3.2 tem-se as ferramentas de software utilizadas. E na seção 3.3 é apresentada a metodologia utilizada para a realização dos experimentos.

Este trabalho tem como objetivo adaptar uma técnica utilizada na IA que aplicada a problemas hierárquicos multirrótulo determina o número de classes que pode ser atribuído a um exemplo real não pertencente à base de treinamento. Para isso a metodologia utilizada no trabalho pode ser dividida em três fases: estudo dos conceitos fundamentais de classificação hierárquica multirrótulo e seus métodos, aplicação dos métodos em diferentes bases de dados, avaliação e análise dos resultados. Cada uma das fases está descrita a seguir:

- **Estudo dos métodos:** Esta etapa tem como início uma grande pesquisa bibliográfica, que teve como objetivo estudar e entender os conceitos referentes à classificação, bem como os métodos utilizados para resolução de problemas de classificação hierárquica multirrótulo.
- **Aplicação dos métodos em diferentes bases de dados:** Nesta fase foi aplicada a predição em bases de funções de proteínas, onde as classes a serem previstas representam funções biológicas de proteínas e são definidas na *Gene Ontology* (GO), mais especificamente a subontologia “função molecular”. Para a utilização das bases foi necessário um pré-processamentos que é apresentado com maior detalhes posteriormente, viabilizando sua utilização pelos algoritmos estudados.
- **Avaliação e análise dos resultados:** Considerando as peculiaridades inerentes aos problemas de classificação hierárquica multirrótulo, medidas especificas para estes tipos de problemas devem ser utilizadas para avaliar os modelos de classificação gerados para solucioná-los. Tais medidas requerem algumas considerações adicionais, além dos aspectos normalmente considerados na avaliação de modelos convencionais de classificação. Na literatura pode ser verificada a utilização de diversas medidas de avaliação para os modelos gerados para estes tipos de

problemas. Essa fase é tratada juntamente com a fase de aplicação dos métodos em diferentes bases de dados.

3.1 BASES DE DADOS

O conjunto de dados utilizados nos experimentos são dados biológicos da área genômica funcional (GO - *Gene Ontology*). Eles foram utilizados por Borges (2012) em sua tese de doutorado. No seu trabalho essas foram normalizadas e tiveram seus atributos faltantes imputados. Elas podem ser obtidos, em seu formato original[1].

A Tabela 1 mostra as principais características das bases de dados usadas nos experimentos. Uma descrição mais detalhada sobre cada uma delas podem ser encontradas em Borges (2012).

Tabela 1 - Características das bases de dados GO

Base de Dados	Quant. Amostras	Quant. Atributos	Quant. Classes	Quant. Max. Níveis	Quant. Min/Max Classes por Amostras	Quant. Min/Max Amostras por Classe
Cellcycle	3751	77	4125	13	3/28	0/785
Church	3749	27	4125	13	3/28	0/786
Derisi	3719	63	4119	13	3/28	0/781
Eisen	2418	79	3573	13	3/28	0/492
Expr	3773	551	4131	13	3/28	0/789
Gasch1	3758	173	4125	13	3/28	0/786
Gasch2	3773	52	4131	13	3/28	0/789
Pheno	1586	69	3127	13	3/28	0/388
Seq	3900	478	4133	13	3/28	0/791
Spo	3697	80	4119	13	3/28	0/775

Fonte: Borges (2012)

[1]http://www.cs.kuleuven.be/~dtai/clus/hmcdatasets

3.2 FERRAMENTAS DE SOFTWARE UTILIZADAS

O *framework* Mulan (TSOUMAKAS *et al.*, 2007) foi utilizado para conduzir os experimentos juntamente com o método de classificação ML-kNN. O Mulan trabalha em conjunto com as classes Java do Weka, um ambiente conhecido e utilizado pela comunidade de aprendizado de máquina e de mineração de dados (Hall *et al.*, 2009; *Witten et al.*, 2011). O pacote contém um *framework* que implementa uma grande gama de métodos multirrótulo, os quais podem ser utilizados e ampliados, bem como disponibiliza diversas medidas de avaliação.

O formato requerido pelo Mulan envolve dois arquivos, dos quais um é do tipo ARFF (*Attribute-Relation File Format*) e o outro é do tipo XML (*eXtensible Markup Language*). Na Figura 13 encontra-se um exemplo do formato de arquivo ARFF aceito pelo Mulan.

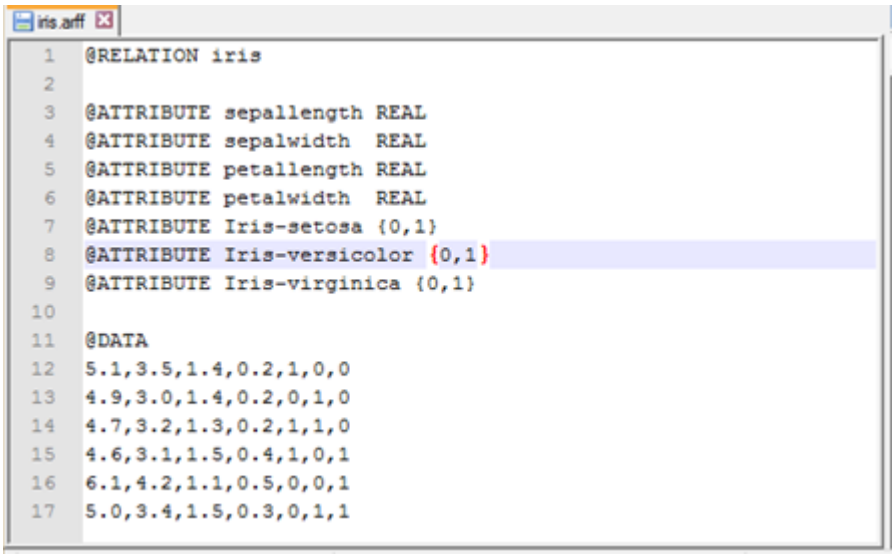

Figura 13 - Exemplo de formato de arquivo ARFF
Fonte: Autoria Própria

No arquivo ARFF é possível definir o tipo de dados que estão sendo carregados, e então fornecer seus próprios dados. No arquivo, foi definido cada coluna e o que cada coluna contém, fornecemos cada linha de dados em um formato delimitado por vírgulas. O arquivo ARFF adotado pelo Mulan possui pequenas diferenças com relação aos arquivos ARFF utilizados em outras ferramentas. Uma diferença é que ao contrário de outras ferramentas, o Mulan não utiliza apenas um atributo do tipo hierárquico para a classe definindo a hierarquia

logo após este atributo. No Mulan cada rótulo vira um atributo classe do tipo {0,1}, onde 1 representa que aquele exemplo é pertencente a classe e 0 a ausência daquela classe.

Na Figura 14 encontra-se um exemplo do formato de arquivo XML aceito pelo Mulan.

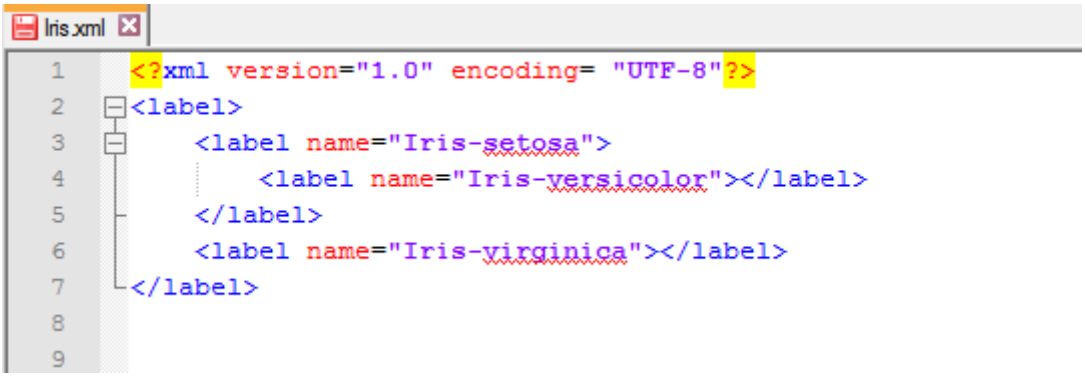

Figura 14 - Exemplo de arquivo XML aceito pelo Mulan
Fonte: Autoria Própria

O arquivo XML é responsável por representar a hierarquia/dependência entre os rótulos/classes da base a ser analisada. Este arquivo se faz necessário durante as etapas de classificação e avaliação.

É importante ressaltar que este arquivo só pode ser utilizado para representar problemas hierárquicos multirrótulo que tenham a hierarquia estruturada em formato do tipo árvore, que são estruturas mais simples, onde um nó filho tem apenas um nó pai.

As bases escolhidas para este trabalho possuem sua hierarquia estruturada em formato do tipo DAG, sendo esta estrutura mais complexa quando comparada ao do tipo árvore, pois um nó filho pode ter múltiplos pais. Devido a esse fato foi necessário criar uma nova forma de representar a hierarquia dos rótulos substituindo assim, a necessidade de fornecer o arquivo do tipo XML como entrada, e permitindo a utilização de qualquer estrutura hierárquica do tipo DAG.

Todas as alterações realizadas neste *framework* serão submetidas à equipe de desenvolvimento para possível incorporação e estarão disponíveis aos pesquisadores da área no próprio site da ferramenta.

3.3 METODOLOGIA UTLIZADA PARA REALIZAÇÃO DOS EXPERIMENTOS

Basicamente a metodologia utilizada para a realização dos experimentos é formada por duas principais etapas: pré-processamento e métodos utilizados conforme pode ser visualizado na Figura 15.

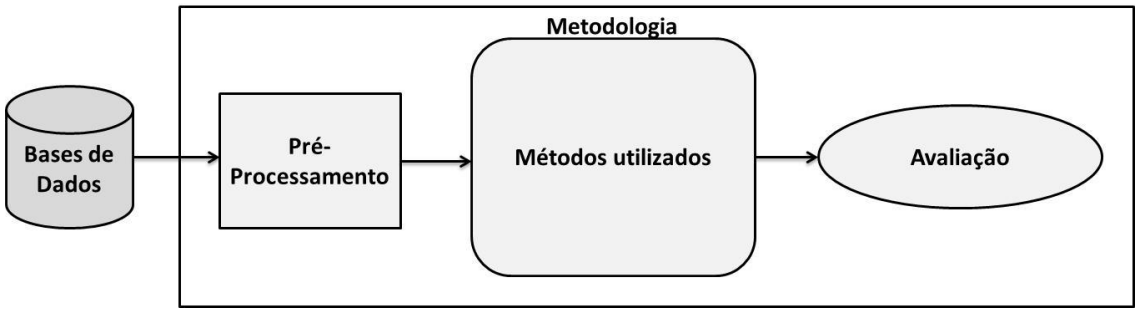

Figura 15 - Metodologia para realização dos experimentos
Fonte: Adaptado de Borges (2012)

3.3.1 Pré-processamento das Bases de Dados

Após a obtenção das bases foi desenvolvido um algoritmo que tem como objetivo principal fazer a adaptação dos arquivos ARFF que são utilizados pelo *framework* Mulan. Este algoritmo é composto das seguintes fases:

- Entrada: O algoritmo recebe dois arquivos de texto. O primeiro arquivo contém a relação de ancestralidade de descendência entre os rótulos, e o segundo possui os valores de cada atributo dos exemplos contidos na base de dados e as devidas classes associadas a cada exemplo.
- Processamento: O primeiro arquivo que é recebido pelo algoritmo, possibilita armazenar em uma lista todos os rótulos existentes na base para que mais tarde, os mesmos sejam consultados na escrita do arquivo de saída. Como dito anteriormente, o Mulan não possui apenas um atributo classe do tipo hierárquico e isso traz impacto no valor atribuído, pois nos outros arquivos as classes são representadas no mesmo atributo e separadas pelo caractere “@”. Para isso, o algoritmo desenvolvido faz a leitura do segundo arquivo de entrada, verifica instância por instância, a qual classe a mesma pertence e define com o valor 1 o atributo que representa aquela classe, sendo que os demais recebem valor 0.

- Saída: Como saída é obtido os arquivos da base de treinamento e teste no formato exigido pelo *framework* Mulan.

3.3.2 Métodos Utilizados

Para a realização dos experimentos com as 10 bases de funções de proteínas foi escolhido o algoritmo ML-kNN (*Multi-label k*NN) que foi criado por Zhang *et al.* (2007) e está implementado no *framework* Mulan. Este *framework* implementa uma grande gama de métodos multirrótulo, os quais podem ser utilizados e ampliados, bem como disponibiliza diversas medidas de avaliação (TSOUMAKAS *et al.*, 2011).

O ML-kNN é um algoritmo conhecido como preguiçoso, pois ele não aprende um modelo compacto de dados, apenas memoriza os exemplos do conjunto de treinamento na memória. Para classificar uma nova instância da base de teste é feita uma comparação dessa instância com todas as instâncias pertencentes ao conjunto de treinamento por meio do cálculo de distância.

O ML-kNN é uma adaptação do algoritmo kNN para o problema multirrótulo. Esse algoritmo determina o conjunto de rótulos do exemplo a ser classificado, baseado na probabilidade máxima a *posteriori* calculada a partir da frequência de cada rótulo entre os k vizinhos mais próximos, comparado com a frequência em todos os exemplos. Pois diferente da classificação unirrótulo a qual pode ser utilizado o algoritmo kNN, no algoritmo ML-kNN, uma instância pode ser classificada com múltiplas classes.

3.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Nesse capítulo foi apresentada a metodologia usada durante o trabalho. O início da pesquisa com o estudo dos conceitos referentes à classificação, bem como os métodos utilizados para resolução de problemas hierárquicos multirrótulo. Foi apresentado também o escopo inicial do programa e o porquê o mesmo teve de ser alterado incorporando novos métodos e uma nova forma de representação da

hierarquia dos rótulos, definindo de maneira detalhada cada fase do desenvolvimento.

4 EXPERIMENTOS E RESULTADOS

Neste capítulo serão apresentados os experimentos realizados e os resultados obtidos com o classificador ML-kNN. Os experimentos foram realizados utilizando dez bases de dados da área genômica funcional *Gene Ontology*, sendo esta estruturada em DAG. Para a avaliação foram utilizadas variações no algoritmo ML-kNN para os valores de k e *threshold*, onde k recebe os valores 3, 5 e 7, enquanto o *threshold* varia entre 0.5, 0.7 e 0.8. Dessa forma, a Seção 4.1faz uma breve descrição sobre as configurações experimentais que foram feitas utilizando o conjunto de bases de dados estruturados na forma de DAG. A Seção 4.2 apresenta os resultados obtidos com os experimentos realizados e a Seção 4.2 apresenta os resultados obtidos no teste estatístico Wilcoxon.

4.1 CONFIGURAÇÕES EXPERIMENTAIS

A fim de ilustrar as modificações propostas, os conjuntos de dados utilizados nos experimentos são dados biológicos da área genômica funcional (GO - *Gene Ontology*). O ML-kNN foi executado variando o parâmetro k que representa o valor dos k-vizinhos mais próximos assumindo os valores de 3, 5 e 7. Também foi variado o valor do *threshold*, sendo este o valor de corte utilizado na comparação com o valor de confiança de cada rótulo de um instância de teste, entre 0.5, 0.7 e 0.8. Para a avaliação do desempenho foram utilizadas as medidas de precisão, revocação e *F-Measure*, sendo estas já descritas nas Equações 2.7, 2.8 e 2.9, respectivamente.

Como já mencionado anteriormente, os experimentos usam as classes implementadas no *framework* Mulan. Sendo utilizada a estratégia de *holdout*, onde a base de dados é dividida em 2/3 para a base de treinamento e 1/3 para a base de teste. Os gráficos foram gerados por meio do *software* Excell e os testes estatísticos de Wilcoxon gerados por um *applet* utilizando o método *Two Paired Sample Signed Rank Test*[2].

[2]http://www.socr.ucla.edu/htmls/ana/TwoPairedSampleSignedRankTest_Analysis.html

4.2 MÉTODO UTILIZADO

Neste tópico será apresentado o algoritmo ML-kNN e as alterações realizadas no mesmo, para que fosse possível realizar a tarefa de predição do número de classes na classificação de dados hierárquicos estruturados, principalmente, na forma de um DAG. Visto que nenhum dos trabalhos encontrados utilizavam bases estruturadas em formato DAG, somente em um tipo especial de DAG, sendo esta estrutura denominada árvore.

O ML-kNN é um algoritmo conhecido como preguiçoso, pois ele não aprende um modelo compacto de dados, apenas memoriza os exemplos do conjunto de treinamento na memória. Para classificar uma nova instância da base de teste é feita uma comparação dessa instância com todas as instâncias pertencentes ao conjunto de treinamento por meio do cálculo de distância.

Possui um funcionamento mais complexo do que o algoritmo kNN tradicional, devido ao fato de que em problemas multirrótulos, os exemplos possuem um conjunto de rótulos e não apenas uma classe mais popular entre os k-vizinhos com menor valor de distância, como acontece nos problemas unirrótulos.

Esse algoritmo determina o conjunto de rótulos do exemplo a ser classificado, baseado na probabilidade máxima a *posteriori* calculada a partir da frequência de cada rótulo entre os k vizinhos mais próximos, comparado com a frequência em todos os exemplos.

Em problemas multirrótulos, há outra diferença, sendo esta relacionada à avaliação da predição de classes, pois em problemas unirrótulos a predição está correta ou errada, e nos problemas multirrótulo devido a dependência dos rótulos, a predição pode estar parcialmente correta, em que alguns dos rótulos são preditos de maneira correta, mas outros não.

Como dito anteriormente, o algoritmo ML-kNN está implementado na ferramenta Mulan, porém esta possui algumas barreiras para a classificação hierárquica multirrótulo, sendo estas citadas no tópico 3.2. Para ultrapassar essas barreiras adotou-se a estratégia de criar um projeto separado do *framework* Mulan, onde seriam adicionadas somente as classes relacionadas ao classificador ML-kNN, tornando assim o desenvolvimento mais fácil e prático, mas ainda respeitando a organização do *framework*, para que esse novo projeto possa ser incorporado futuramente ao mesmo.

A primeira alteração realizada foi criar um novo formato de arquivo de entrada para representar a hierarquia dos rótulos, visto que o arquivo XML do Mulan só era capaz de representar bases estruturadas no formato tipo árvore. Optou-se então, por receber como entrada no algoritmo um arquivo de texto que continha as informações de dependência entre os rótulos, no mesmo formato que pode ser observado na Figura 16.

```
1  RAIZ 1
2  1 2
3  1 3
4  1 7
5  2 4
6  2 5
7  3 5
8  3 6
9  7 6
10 7 9
11 6 8
```

Figura 16 - Exemplo de arquivo de texto com informações de dependência entre os rótulos
Fonte: Autoria Própria

Através desse arquivo, é possível montar a estrutura de um grafo acíclico direcionado, mais conhecido, como DAG, pois é possível armazenar a lista com o nome dos vértices e seus k-vizinhos. Este arquivo preserva a dependência entre os rótulos, sendo este um fator importante tanto para a classificação quanto para a realização dos cálculos na avaliação do desempenho do algoritmo.

O arquivo ARFF exigido pelo Mulan ainda é mantido, e para a realização dos experimentos são utilizadas as bases geradas na etapa de pré-processamento. Ao garantir os arquivos de entrada, a próxima alteração realizada foi criar novos métodos nas classes importadas do Mulan, sendo estas: *MLkNN*, *MultiLabelKNN*, *MultiLabelLearner*, *MultiLabelLearnerBase*e *MultiLabelOutput*.

A criação de novos métodos se fez necessária devido ao fato da modificação da estrutura que o arquivo XML representava e era armazenada, sendo agora representada por um arquivo de texto e armazenada em um grafo.

A leitura dos arquivos da base permanece da mesma maneira, sendo utilizada a biblioteca Weka para interpretar os arquivos do formato ARFF. Após a leitura das bases é instanciado o classificador ML-kNN, passando como parâmetro o valor de k-vizinhos e o fator de suavização padrão para a classificação.

Para a construção da etapa de treinamento, é passado para o classificador a base de treinamento e após esse processo, é realizada a predição, sendo esta realizada como citado anteriormente, a cada nova instância é calculada a distância euclidiana dela com cada instância da base de treinamento. Para a predição foram realizadas algumas mudanças, a primeira delas foi estabelecer um valor para o *threshold* (valor de corte).

Para melhorar o funcionamento do algoritmo com relação à predição, utilizou-se a técnica de Spyromitros *et al.* (2008). Neste trabalho a técnica que antes era utilizada no algoritmo de transformação BR-kNN será adaptada para o algoritmo ML-kNN. Esta técnica consiste em considerar um valor de confiança de cada rótulo para decidir os que farão parte do multirrótulo predito, a saber: a média do número de ocorrências dos rótulos nos k exemplos.

Partindo desse princípio, a seleção dos rótulos é feita através da comparação entre o valor da confiança do rótulo com o valor do *threshold*, caso a confiança seja superior ao *threshold* estipulado, o rótulo é considerado parte do multirrótulo predito. Entretanto, esse procedimento de seleção de rótulos pode predizer multirrótulos vazios. Com o objetivo de anular essa possibilidade, caso nenhum rótulo seja selecionado, o de maior confiança é considerado parte do multirrótulo pelo ML-kNN.

Após a predição das classes, é realizada uma busca no grafo que representa a hierarquia das classes, com o intuito de atribuir valor 1 para os atributos ancestrais ao nó predito. Este mesmo processo de atribuir o valor 1 para os atributos ancestrais é realizado com os atributos-classe da base de teste.

Para avaliar o desempenho do algoritmo ML-kNN foram implementadas as medidas de precisão e revocação, além de ser realizada a variação do valor de k e do *threshold*.

4.3 CLASSIFICAÇÃO HIERÁRQUICA MULTIRRÓTULO

Nessa seção serão apresentados os resultados experimentais obtidos após a aplicação do algoritmo de classificação em bases de dados hierárquico multirrótulo.

As medidas de avaliação foram calculadas para cada base realizando todas as combinações possíveis dos valores de k e *threshold,* com o intuito de observar o comportamento do processo de predição do número de classes, visto que os valores de k e *threshold* influenciam diretamente no processo.

Essas medidas hierárquicas levam em consideração não apenas o nó predito, mas também todos os seus ancestrais, pois uma predição em problemas multirrótulos pode estar correta, errada ou parcialmente correta. As medidas aplicadas foram a de Precisão Hierárquica, Revocação Hierárquica e *F-Measure* Hierárquica.

Nas Tabelas 2, 3 e 4, estão tabulados os valores obtidos através do cálculo das medidas de precisão e revocação para cada base. Para cada tabela foi mantido um valor de *threshold* e varia-se os valores de k para 3, 5 e 7.

Tabela 2 - Resultados dos experimentos para *threshold* = 0.5

	Threshold = 0.5					
	K = 3		**K = 5**		**K = 7**	
	Prec.	**Rev.**	**Prec.**	**Rev.**	**Prec.**	**Rev.**
Cellcycle	0,7359	0,2966	0,7589	0,2888	0,7467	0,2968
Church	0,7687	0,248	0,7721	0,2469	0,7353	0,2631
Derisi	0,7568	0,2573	0,7527	0,2623	0,7606	0,261
Eisen	0,7432	0,2981	0,7406	0,301	0,7445	0,2957
Expr	0,771	0,2926	0,7675	0,2998	0,7597	0,3048
Gasch1	0,7627	0,2915	0,7609	0,2968	0,761	0,2985
Gasch2	0,7474	0,2868	0,7624	0,2778	0,7708	0,274
Pheno	0,7654	0,2441	0,7329	0,2547	0,7431	0,2485
Seq	0,771	0,2864	0,7761	0,2828	0,7797	0,283
Spo	0,741	0,2794	0,7358	0,285	0,7448	0,2802

Fonte: Autoria Própria

Tabela 3 - Resultados dos experimentos para *threshold* = 0.7

	Threshold = 0.7					
	K = 3		K = 5		K = 7	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
Cellcycle	0,8986	0,2006	0,8986	0,209	0,9104	0,2013
Church	0,8972	0,1876	0,895	0,1887	0,8973	0,188
Derisi	0,895	0,1908	0,8922	0,1923	0,893	0,1935
Eisen	0,8545	0,2179	0,8707	0,2136	0,8821	0,2075
Expr	0,8841	0,2286	0,8869	0,2282	0,8973	0,2208
Gasch1	0,8809	0,2192	0,8815	0,2287	0,8931	0,2187
Gasch2	0,8985	0,1989	0,8949	0,2071	0,9005	0,2045
Pheno	0,8936	0,1847	0,8922	0,1861	0,8866	0,1865
Seq	0,9121	0,2013	0,9056	0,2062	0,9	0,212
Spo	0,9019	0,1941	0,8897	0,2043	0,8889	0,2005

Fonte: Autoria Própria

Tabela 4 - Resultado dos experimentos para *threshold* = 0.8

	Threshold = 0.8					
	K = 3		K = 5		K = 7	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
Cellcycle	0,9175	0,1871	0,9245	0,1824	0,9292	0,1834
Church	0,9464	0,1539	0,9467	0,1538	0,9464	0,1538
Derisi	0,9427	0,1563	0,943	0,1554	0,9364	0,1632
Eisen	0,9234	0,1663	0,9056	0,1856	0,9101	0,1857
Expr	0,9243	0,1935	0,9281	0,1948	0,9334	0,1959
Gasch1	0,9201	0,1873	0,9208	0,1951	0,9256	0,1916
Gasch2	0,9195	0,1833	0,9275	0,1811	0,933	0,1792
Pheno	0,9258	0,1604	0,9367	0,1512	0,9296	0,1569
Seq	0,9279	0,1814	0,9179	0,1941	0,9099	0,1934
Spo	0,9398	0,1576	0,926	0,174	0,9276	0,1713

Fonte: Autoria Própria

Através dos valores das medidas de avaliação encontradas nas Tabelas 2, 3 e 4, pode-se perceber que tanto a medida de precisão quanto a medida de revocação tem mudanças poucas significativas, quando levado em consideração apenas a variação dos valores de k. Ao contrário do que acontece com a variação do *threshold*, quanto maior o valor atribuído ao *threshold*, maior é o valor encontrado para a medida de precisão.

Após os cálculos das medidas de precisão e revocação, é calculada a medida *F-Measure*, sendo esta a medida harmônica das outras duas. Para isso utilizou-se a ferramenta Excel para gerar os valores de todas as bases. Para melhor visualização estas medidas foram plotadas nos Gráficos 1, 2 e 3.

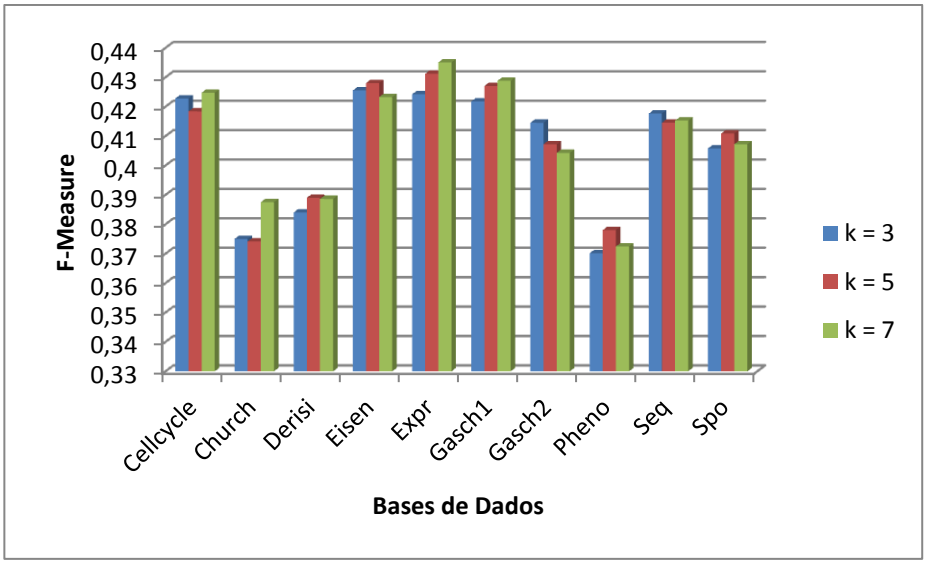

Gráfico 1 - Medida *F-Measure* com *threshold* 0.5
Fonte: Autoria Própria

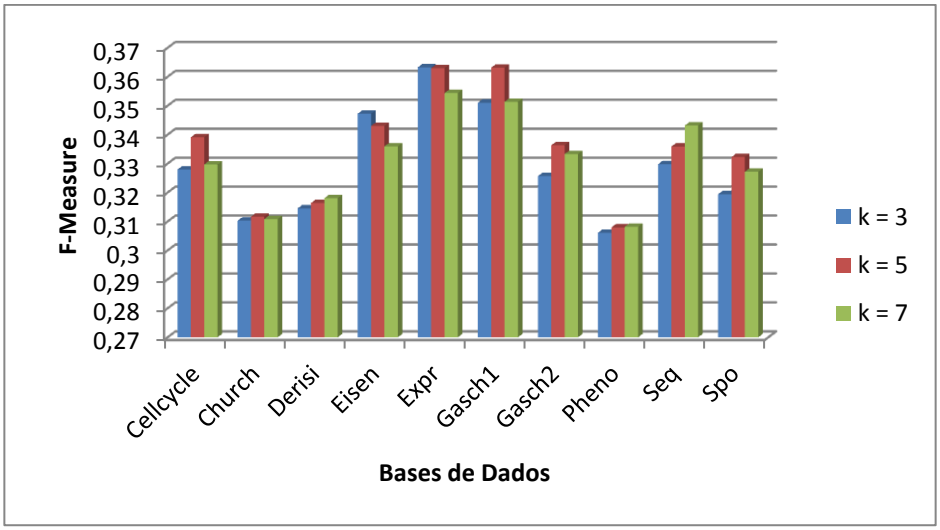

Gráfico 2 - Medida *F-Measure* com *threshold* 0.7
Fonte: Autoria Própria

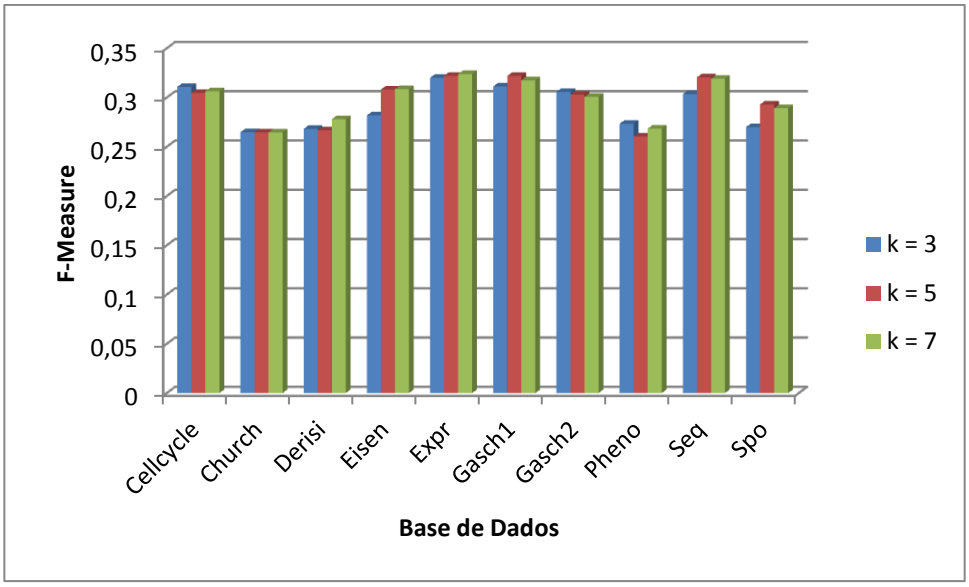

Gráfico 3 - Medida *F-Measure* com *threshold* 0.8
Fonte: Autoria Própria

Através dos gráficos é possível perceber que na mesma base, a variação da medida de *F-Measure* é pouco significativa. E que o seu valor é próximo de 0, demonstrando que quando comparadas as medidas de precisão e revocação, o algoritmo se mostra pouco eficaz.

4.4 AVALIAÇÃO ESTATÍSTICA

Nesta seção são apresentados os resultados obtidos na realização do teste estatístico de Wilcoxon. Para realizar esse teste estatístico foi escolhido os valores da medida de precisão. Na seção 4.3.1 são apresentados os resultados obtidos de z levando-se em consideração o valor do *threshold* como constante. Na seção 4.3.2 serão apresentados os resultados obtidos de z levando-se em consideração o valor de k como constante.

Para as duas variações realizadas do teste de Wilcoxon, assim como mencionado no tópico 2.5.3.1 a hipótese nula assume que a diferença de desempenho entre algoritmos não é significativa. Com nível de confiança $\alpha = 0.05$a hipótese nula não pode ser rejeitada se $-1.96 \leq z \leq 1.96$. Para a realização dos cálculos foi utilizado um *applet* assim como citado no tópico 4.1 onde é informado os dois conjuntos de valores a serem comparados pelo teste, e como saída desse programa é gerado o valor de z.

4.4.1 Teste de Wilcoxon com Valor de Threshold Constante

Nesta seção encontram-se os resultados obtidos pelo Teste de Wilcoxon levando em consideração a variação do valor de k. Nas Tabelas 5, 6 e 7 são encontrados os valores tabulados do teste para k assumindo os valores 3, 5 e 7.

Tabela5 – Teste Wilcoxon *threshold* 0.5

Threshold = 0.5	
K = 3 com k =5	0.153
K = 3 com k = 7	0.051
K = 5 com k = 7	0.459

Fonte: Autoria Própria

Tabela6 - Teste Wilcoxon *threshold* 0.7

Threshold = 0.7	
K = 3 com k =5	0.948
K = 3 com k = 7	0.714
K = 5 com k = 7	1.631

Fonte: Autoria Própria

Tabela7 - Teste Wilcoxon *threshold* 0.8

Threshold = 0.8	
K = 3 com k =5	0.153
K = 3 com k = 7	0.296
K = 5 com k = 7	0.051

Fonte: Autoria Própria

Através dos valores de z encontrados nas Tabelas 5, 6 e 7 pode-se concluir que nenhum valor permite que a hipótese nula seja rejeitada, ou seja, não há melhora no desempenho do algoritmo realizando a variação dos valores de k.

4.4.2 Teste de Wilcoxon com Valor de k Constante

Nesta seção encontram-se os resultados obtidos pelo Teste de Wilcoxon levando em consideração a variação do valor de *threshold*. Nas tabelas X. Y. Z são

encontrados os valores tabulados do teste para *threshold* assumindo os valores 0.5. 0.7 e 0.8.

Tabela8 – Teste Wilcoxon k = 3

K = 3	
Threshold = 0.5 com 0.7	Z-Score = -2.8031
Threshold = 0.5 com 0.8	Z-Score = -2.8031
Threshold = 0.8 com 0.7	Z-Score = -2.8031

Fonte: Autoria Própria

Tabela 9 - Teste Wilcoxon k = 5

K = 5	
Threshold = 0.5 com 0.7	Z-Score = -2.8030
Threshold = 0.5 com 0.8	Z-Score = -2.8030
Threshold = 0.8 com 0.7	Z-Score = -2.8030

Fonte: Autoria Própria

Tabela10 – Teste Wilcoxon k = 7

K = 7	
Threshold = 0.5 com 0.7	Z-Score = -2.8032
Threshold = 0.5 com 0.8	Z-Score = -2.8032
Threshold = 0.8 com 0.7	Z-Score = -2.8032

Fonte: Autoria Própria

Ao contrário do resultado obtido na variação dos valores de k, é possível perceber que neste teste, os valores obtidos de z, são superiores ao limite de -1.96, o que significa que a hipótese nula pode ser rejeitada e que há melhoras no desempenho do algoritmo.

4.5 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capítulo foi apresentado os experimentos realizados a fim de ilustrar as modificações realizadas no algoritmo Ml-kNN. Os experimentos foram realizados utilizando dez bases de dados da área genômica funcional, *Gene Ontology*, sendo estas estruturas em DAG. Para a avaliação foram utilizadas variações no algoritmo ML-kNN para os valores de *k* e *threshold*, onde *k* recebe os valores 3, 5 e 7, enquanto o *threshold* varia entre 0.5, 0.7 e 0.8.

No capítulo foi feita uma breve descrição sobre as configurações experimentais que foram feitas utilizando o conjunto de bases de dados estruturados na forma de DAG. Bem como a descrição detalhada de todo o processo de alteração no algoritmo ML-kNN. Foram apresentados também os resultados obtidos com os experimentos e os resultados obtidos através dos testes estatísticos Wilcoxon.

5 CONCLUSÃO

Neste capítulo são destacadas as considerações finais do trabalho, apresentando algumas sugestões para trabalhos futuros.

5.1 CONSIDERAÇÕES FINAIS

Esta seção apresenta as considerações finais sobre o estudo, confrontando os objetivos gerais e específicos com os resultados obtidos através dos experimentos realizados.

O objetivo geral de aplicar uma técnica para predizer a quantidade de classes em problemas hierárquicos multirrótulos foi alcançado utilizando a metodologia desenvolvida. A metodologia engloba as seguintes fases: pré-processamento das bases de dados, adaptação do algoritmo de classificação hierárquico multirrótulo ML-kNN, aplicação do algoritmo nas bases de dados e avaliação estatística dos resultados.

Durante o desenvolvimento do trabalho, foram necessárias algumas mudanças para utilizar o algoritmo de classificação multirrótulo ML-kNN. Dentre essas mudanças, pode-se citar a modificação do arquivo de entrada XML do *framework* Mulan, onde encontra-se implementado o algoritmo. Pode-se ressaltar também o fato da utilização da técnica antes utilizada apenas no algoritmo BR-kNN, se na fase de predição não for atribuída nenhuma classe a instância, atribuísse a essa instância então a classe que possuir o maior valor de confiança.

Quanto ao objetivo específico de estudar classificação hierárquica, problemas multirrótulo e técnicas utilizadas para predição em problemas multirrótulos, o referencial teórico levantado sobre os assuntos mostra a validade da pesquisa.

O objetivo específico de realizar experimentos com o algoritmo adaptado foi alcançado, visto que os resultados dos experimentos sugerem que o algoritmo ML-kNN tem um desempenho superior quando relacionada à medida de precisão aos demais experimentos quando assume valor da variável de corte igual a 0.8, esse fato pode ser explicado na comparação entre o valor da confiança dos rótulos com o

threshold, pois quanto maior o valor do *threshold* a tendência é que seja predito um número cada vez menor de classes, entrando na condição onde é atribuída apenas a classe com a maior confiança dentre os rótulos daquela instância. A partir disso, sabe-se que a precisão é calculada pelo número de classes da intercessão entre verdadeiras e preditas divididas pelo número de classes preditas, como nesse caso o número de preditas é 1, a tendência é que a precisão seja maior.

Assim é possível concluir que esse trabalho cumpriu com todos os objetivos propostos, além de ser considerado como sendo um dos primeiros a tratar bases hierárquicas multirrótulo em formato DAG utilizando o *framework* Mulan.

5.2 TRABALHOS FUTUROS

Com base nos resultados obtidos, algumas sugestões para trabalhos futuros podem ser listadas:

- Avaliação do algoritmo ML-kNN com outros conjuntos de dados, de modo a fortalecer a avaliação experimental.
- Utilização de outro algoritmo de classificação multirrótulo para realizar a comparação do seu desempenho de funcionamento com o algoritmo apresentado nesse trabalho;
- Utilizar outras medidas de avaliação;
- Utilizar o projeto proposto alterando os valores dos parâmetros de k e do *threshold*.

REFERÊNCIAS

ALVES, R. T. **Um Sistema Imunológico Artificial para Classificação Hierárquica e Multi-Label de Funções de Proteínas.** 2010. 219 p. Tese (Doutorado) - Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial. Universidade Tecnológica Federal do Paraná (UTFPR), Curitiba, Paraná, 2010.

BARUTCUOGLU, Z.; SCHAPIRE, R. E.; TROYANSKAYA, O. G. Hierarchical multi-label prediction of gene function. **In:** Bioinformatics, v. 22. n. 7. p. 830, 2006.

BLOCKEEL, H.; SCHIETGAT, L.; STRUYF, J.; DZEROSKI, S.; CLARE, A. **Decision trees for hierarchical multilabel classification: A case study in functional genomics.** PKDD, v. 4213, 2006.

BORGES, H. B. **Classificador Hierárquico Multirrótulo Usando uma Rede Neural Competitiva.** 2012. 188 p. Tese (Doutorado) – Programa de Pós-graduação em Informática. Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Paraná, 2012.

BOUTELL, M. R.; LUO, J.; SHEN, X.; BROWN, C. M. Learning Multi-Label Scene Classification. **Pattern Recognition**, Rochester, 4 marc. 2004.

CARVALHO, A. C. P. F.; FREITAS, A. A. **Tutorial on Hierarchical Classification with Applications in Bioinformatics.**v.1.São Paulo: Idea Group, 2007.

CERRI, R.; CARVALHO, A. C. P. L. F. **Técnicas de Classificação Hierárquica Multirrótulo.** 2010. 241 p. Dissertação (Mestrado) – Instituto de Ciências Matemáticas e de Computação (ICMC-USP). Universidade de São Paulo, São Carlos, São Paulo, 2010.

CESA-BIANCHI, N.; GENTILE, C.; ZANIBONI, L. Incremental algorithms for hierarchical classification. **Journal of Machine Learning Research**, Italia, p. 31, jan. 2006.

CHU, S.; DERISI, J.; EISEN, M.; MULHOLLAND, J.; BOTSTEIN, D.; BROWN, P.; HERSKOWITZ, I.The Transcriptional Program of Sporulation in Budding Yeast. **Science**, Washington, dez. 1998, p.699.

CLARE, A.; KING, R. D. Knowledge Discovery in Multi-Label Phenotype Data. **In:** Principles of Data Mining and Knowledge Discovery. Friburgo, Alemanha, 2001.

CLARE, A. **Machine Learning and Data Mining for Yeast Functional Genomics.**2003.210 p. Tese (Doutorado) – Pós-graduação em Filosofia. The University of Wales, Aberystwyth, Reino Unido, 2003.

COSTA, E. **Investigação de Técnicas de Classificação Hierárquica para Problemas de Bioinformática**. 2008. 184 p. Dissertação (Mestrado) – Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2008.

DESMAR, J. Statistical comparions of classifiers over multiple data sets. **Jornal of Machine Learning Research.**v. 7, p. 1-30, 2006.

EISEN, M.; SPELLMAN, P.; BROWN, P.; BOTSTEIN, D. Cluster Analysis and Display of Genome-Wide Expression Patterns.**Proceedings of the National Academy of Sciences.** EstadosUnidos, v. 95, dez. 1998. Disponível em: <http:// www.pnas.org/content/95/25/14863.full.pdf+html>. Acessoem: 25 nov. 2013.

ELISSEEF, A.; WESTON, J. Kernel Methods for Multi-Labelled Classification and Categorical Regression Problems. **In:** In Proceedings of the 13th International Symposium on String Processing and Information Retrieval, Nova Iorque, 2006.

ESULI, A.; FAGNI, T.; SEBASTIANI, F. Treeboost. Mh: A Boosting Algorithm for Multi-Label Hierarchical Text Categorization. **In**: In Proceedings of the 13th International Symposium on String Processing and Information Retrieval (SPIRE06), Italia, 2006.

FIX, E.; HODGES, J. Discriminatory Analysis, Non-Parametric Discrimination: Consistency Properties. **The Annals of Mathematical Statistics.** v. 36. n.3. p. 1049, jun. 1951.

FREUND, Y.; SCHAPIRE, R. E.A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. **In**: European Conference on Computational Learning Theory, Nova Jersey, 1995.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. **In:** Annals of Mathematical Statistics. v. 11, p. 86-92, 1940.

GASCH, A.; HUANG, M.; METZNER, S.; BOTSTEIN, D.; ELLEDGE, S.; BROWN, P. Genomic Expression Responses to Dna-Damaging Agents and the Regulatory Role of the Yeast Atr Homolog Meclp. **Molecular Biology of the Cell.**V.12, n.10, p.2987-3000, 2001.

GASCH, A.; SPELLMAN, P.; KAO, C.; CARMEL-HAREL, O.; EISEN, M.; STORZ, G.; BOTSTEIN, D.; BROWN, P. Genomic expression programs in the response of yeast cells to environmental changes. **Molecular Biology of the Cell.**V.12, p. 4241-4257, 2000.

GASTEIGER, E.; HOOGLAND, C.; GATTIKER, A.; DUVAUD, S.; WILKINS, M.; APPEL, R.; BAIROCH, A. Protein identification and analysis tools on the expasy server. **In:** The Proteomics Protocols Handbook, Humana Press, 2005.

GHAMRAWI, N.; MCCALLUM, A. Collective multi-label classification. **In:** Proceedings of the 14th ACM International Conference on Information's and Knowledge Management. Nova Iorque, EUA, 2005.

GONÇALVES, T.; QUARESMA, P.A preliminary approach to the multilabel classification problem of Portuguese juridicial documents. **In:** EPIA, 2003.

HALL M.; FRANK, E.; HOLMES, G.; PFAHRINGER, B.; REUTEMANN, P.; WITTEN, I. H.. 2009. The WEKA data mining software: an update. **SIGKDD Explor. News**, v. 11, p.10-18, 2009.

HAYKIN, S. Redes neurais: princípios e prática. 2.ed. Tradução de, Paulo Martins Engel. Porto Alegre: Bookman, 2001.

IMAN, R. L.; DAVENPORT, J. M. Approximations of the critical region of the friedman statistic. **Communications in Statistics.** p. 571-595, 1980.

KARALIC, A.; PIRNAT, V. Significance level based multiple tree classification. **In:** Informatica, 1991.

KATAKIS, I.; TSOUMAKAS, G.; VLAHAVAS, I. Multilabel text classification for automated tag suggestion. **In:** Proceedings of the ECML/PKDD-08 Workshop on Discovery Challenge, 2008.

KIRITCHENKO, S.; MATWIN, S.; FAMILI, A. F. Hierarchical text categorization as a tool of associating genes with gene ontology codes. **In:** Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics, Pisa, Italia, 2004.

KUMAR, A.; CHEUNG, K.; ROSS-MACDONALD, P.; COELHO, P.; MILLER, P.; SNYDER, M. Triples: a database of gene function in saccharomyces cerevisiae. **In:** Nucleic Acids Research, 2000.

LAUSER, B.; HOTHO, A. Automatic multi-label subject indexing in a multilingual environment. **In**: Proc. of the 7th European Conference in Research and Advanced Technology for Digital Libraries, ECDL, 2003.

LUO, X.; ZINCIR-HEYWOOD, N. A. Evaluation of two systems on multi-class multi-label document classification. **In:** International Symposium on Methodologies for Intelligent Systems, 2005.

MATSUBARA, E. T.; MONARD, M. C.; BATISTA, G. E. A. P. A. **Multi-view semi-supervised learning:** Na approach to obtain different views from text datasets. Himeji, Japão: IOS Press, 2005.

MEWES, H. W.; FRISHMAN, D.; MAYER, K. F. X.; MUNSTERKOTTER, M.; NOUBIBOU, O.; RATTEI, T.; OESTERHELD, M.; STUMPFLEN, V. Mips: Analysis and annotation of proteins from whole genomes. **In:** Nucleic Acids Res, 2004.

MCCALLUM, A. K. Multi-label text classification with a mixture model trained. **In:** 99 Workshop on Text Learning. Pittsburg, 1999.

MITCHELL, T. M. **Machine Learning.** Nova Iorque: McGraw-Hill Higher Education. 1997.

MONARD, M. C.; BARANAUSKAS, J. A. **Conceitos sobre Aprendizado de Máquina.** Sistemas Inteligentes: Fundamentos e Aplicações. São Carlos: Manole, 2003.

OTERO, F. E. B.; FREITAS, A. A.; JOHNSON, C. G. A hierarchical multi-label classification ant colony algorithm for protein function prediction. **In:** Proceedings of the IEEE Conference on Signal Processing, Communication and Applications (SIU'2008), v. 2, p. 165, 2010.

OLIVER, S.A network approach to the systematic analysis of yeast gene function. **In:** Trends in genetics: TIG, 1996.

REZENDE, S, O. **Sistemas Inteligentes:** Fundamentos e Aplicações. Barueri, São Paulo: Manole, 2005.

ROTH1JT, F.; HUGHES, J.; ESTEP, p.; CHURCH, G. Finding dna regulatory motifs within unaligned noncoding sequences clustered by whole-genome mrna quantitation. **In:** Nature biotechnology, 1998.

ROUSU, J.; SAUNDERS, C.; SZEDMAK, S.; SHAWE-TAYLOR, J. Kernel-based learning of hierarchical multilabel classification models. **Journal of Machine Learning Research**, v. 7, p.1601, 2006.

SANTOS, A. M. **Investigando a Combinação de Técnicas de Aprendizado Semissupervisionado e Classificação Hierárquica Multirrótulo.** 2012. 214 p. Tese (Doutorado) – Programa de Pós-Graduação em Sistemas e Computação. Universidade Federal do Rio Grande do Norte (UFRN), Natal, 2012.

SCHAPIRE, R.; SINGER, Y. Improved boosting algorithms using confidence-rated predictions. **In:** Machine Learning, Massachusetts, EUA, v. 37, 1999.

SCHAPIRE, R.; SINGER, Y. Boostexter: A boosting-based system for text categorization. **In:** Machine Learning, Massachusetts, EUA, v. 39, 2000.

SHEN, X.; BOUTELL, M.; LUO, J.; BROWN, C. Multi-label machine learning and its application to semantic scene classification. **In:** International Symposium on Electronic Imaging, San Jose, Califórnia, 2004.

SPELLMAN, P.; SHERLOCK, G.; ZHANG, M.; IYER, V.; ANDERS, K.; EISEN, M.; BROWN, P.; BOTSTEIN, D.; FUTCHER, B. Comprehensive identification of cell cycle regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. **In:** Molecular biology of the cell, 1998.

SPYROMITROS, E.; TSOUMAKAS, G.; VLAHAVAS, I. An empirical study of lazy multilabel classification algorithms. **In**: Hellenic conference on Artificial Intelligence,p. 401–406, Berlin, Alemanha, 2009.

STRUYF, J.; BLOCKEEL, H.; CLARE, A. Hierarchical multi-classification with predictive clustering trees in functional genomics. **In:** Workshop on Computational Methods in Bioinformatics at the 12th Portuguese Conference on Artificial Intelligence. Berlim, Alemanha, 2005.

SUN, A.; LIM, E. P. Hierarchical text classification and evaluation. **In:** Fourth IEEE International Conference on Data Mining, p. 521, 2001.

TAN, P. N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining.** Boston, EUA, Addison-Wesley Longman Publishing CO, 2005.

TROHIDIS, K.; TSOUMAKAS, G.; VLAHAVAS, I. Multilabel classification of music into emotions. **In:** Proc. 9th International Conference on Music Information Retrieval, Filadélfia, EUA, 2008

TSOUMAKAS, G.; KATAKIS, I. Multi label classification: An overview. **International Journal of Data Warehousing and Mining**,13 set. 2007. Idea Group Publishing, 2007.

TSOUMAKAS, G.; KATAKIS, I.; VLAHAVAS, I. Random k-labelsets for multi-label classification. **In:** IEEE Transactions on Knowledge and Data Engineering, IEEE Computer Society, Los Alamitos, EUA, 2010.

TSOUMAKAS, G., XIOUFIS, E. S., VILCEK, J., AND VLAHAVAS, I. P. Mulan: A java library for multi-label learning. **Journal of Machine Learning Research**, 7 jul. 2011.

VALENTINI, G. True path rule hierarchical ensembles. **In:** IEEE/ACM Transactions on Computational Biology and Bioinformatics, v. 5519, 8 jun. 2011.

VENS, C.; STRUYF, J.; SCHIETGAT, L.; DZEROSKI, S.; BLOCKEEL, H. Decision trees for hierarchical multi-label classification. **In:** Machine Learning, 2008.

WILCOXON, F. Individual comparions by ranking methods.**Biometrics.**v.1, p. 80-83, 1945.

YANG, Y.An evaluation of statistical approaches to text categorization. **In:** Information Retrieval, 2005.

YOUNES, Z.; ABDALLAH, F.; DENOEUX, T.; SNOUSSI, H. A dependent multilabel classification method derived from the k-nearest neighbor rule. **EURASIP Journal on Advances in Signal Processing**, Troyes, França, 21 fev. 2011. Hindawi Publishing Corporation, p.14.

ZHANG, M. L.; ZHOU, Z. H. Ml-knn: A lazy learning approach to multi-label learning. **Pattern Recognition**, China, 2007, p. 21.