Lucas Henrique Sousa Mello

UMA ANÁLISE EXPERIMENTAL DE MÉTODOS PARA CLASSIFICAÇÃO MULTIRRÓTULO

Vitória - ES, Brasil

9 de Junho de 2014

Lucas Henrique Sousa Mello

UMA ANÁLISE EXPERIMENTAL DE MÉTODOS PARA CLASSIFICAÇÃO MULTIRRÓTULO

Monografia apresentada para obtenção do Grau de Bacharel em Ciência da Computação pela Universidade Federal do Espírito Santo.

Orientador:
Flávio Miguel Varejão

Departamento de Informática
Centro Tecnológico
Universidade Federal do Espírito Santo

Vitória - ES, Brasil

9 de Junho de 2014

Monografia de Projeto Final de Graduação sob o título *"UMA ANÁLISE EXPERIMENTAL DE MÉTODOS PARA CLASSIFICAÇÃO MULTIRRÓTULO"*, defendida por Lucas Henrique Sousa Mello e aprovada em ???, em Vitória, Estado do Espírito Santo, pela banca examinadora constituída pelos professores:

Prof. Dr. Flávio Miguel Varejão
Departamento de Informática - UFES
Orientador

Prof. Dr. Thomas Walter Rauber
Departamento de Informática - UFES

Prof. Dr. Elias Silva de Oliveira
Departamento de Informática - UFES

Resumo

O problema de classificação de objetos ou dados está presente em diversas aplicações. Problemas de classificação em sua forma mais simples, cada objeto ou amostra específica de dados está associado a uma única classe dentre duas possíveis. Esses problemas são comumente chamados de classificação binária onde uma das classes é chamada de positiva e a outra de negativa. Já na classificação unirrótulo, também chamada de classificação multiclasse, cada objeto está associado a uma dentre várias possíveis classes. Uma forma mais complexa que a classificação unirrótulo, são os problemas de classificação multirrótulo onde objetos podem estar associados a mais de uma classe simultaneamente. Nesse último, o desenvolvimento de algoritmos para resolvê-lo se torna um desafio muito maior do que os anteriores pelo fato de que o número de possíveis respostas passa a crescer exponencialmente, ao invés de linearmente como na classificação unirrótulo. Com isso, métodos como o *Binary Relevance*, *Dependent Binary Relevance* (DBR), *Classifier Chain*, *Ensemble of Classifier Chain* e *Monte Carlo Classifier Chain* são propostos a resolver esse problema. O presente trabalho analisa a aplicação dos métodos citados bem como apresenta o desenvolvimento de um novo método, denominado de Recursive Dependent Binary Relevance que é uma otimização do *Dependent Binary Relevance* (DBR). Análises baseadas em experimentos mostram que o método Recursive Dependent Binary Relevance supera o DBR para várias medidas de qualidade de desempenho. Adicionalmente, as análises mostram quais métodos exploram a correlação entre os rótulos.

Conteúdo

Lista de Tabelas

1 Introdução

Segundo (REZENDE, 2003) Aprendizado de Máquina é uma área da Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Dentro dessa área, encontra-se a subárea Aprendizado Supervisionado. Em Aprendizado Supervisionado, um problema de classificação é a tarefa de encontrar uma técnica capaz de predizer a classe ou as classes que uma instância pertence (REZENDE, 2003). Uma instância é um objeto do mundo real descrito por um vetor de valores numéricos ou nominais e por um conjunto de rótulos. Para completar essa tarefa, a técnica deve usar exemplos de treino cujas classes são conhecidas. Na literatura (REZENDE, 2003) as classes são também chamadas de rótulos e quando as instâncias só podem assumir um único rótulo, o problema é chamado de classificação unirrótulo, do contrário, é chamado de problema de classificação multirrótulo (BORGES, 2012).

Problemas de classificação multirrótulo estão presentes em diversas áreas, trabalhos relevantes podem ser encontrados em áreas como a bioinfomática, diagnóstico médico, classificação de imagens e principalmente categorização de textos, conforme (CARVALHO; FREITAS, 2009). A classificação multirrótulo é inevitavelmente mais complexa que a unirrótulo. Para solucioná-la, o método multirrótulo mais conhecido é um método simples chamado de Relevância Binária (*Binary Relevance - BR*) (CARVALHO; FREITAS, 2009). No entanto, há muitas críticas sobre o *BR*, sendo a maior delas a incapacidade do método de reconhecer a correlação entre os rótulos, como dito por (DEMBCZYNSKI; CHENG; HÜLLERMEIER, 2010). Com o intuito de alcançar melhores resultados que o *BR*, alguns autores, como (READ et al., 2009) e (MONTANES et al., 2014), o aprimoraram ou elaboraram novos tipos de métodos baseados nele, os quais procuram explorar a dependência entre os rótulos.

Com tantos métodos novos, alguns deles apresentados por (CARVALHO; FREITAS, 2009) e por (READ et al., 2009) é necessário realizar comparações e testes de qualidade. É certo que já existem análises e comparações entre os métodos, no entanto há necessidade

de avaliar os métodos mais formalmente e reforçar as conclusões alcançadas pelos autores dos métodos.

1.1 Motivações

O melhor entendimento do funcionamento dos métodos multirrótulo permite:

- descobrir atributos destes que se alterados, aproveitados e/ou combinados podem acarretar na criação de novos métodos e/ou na melhora dos existentes.
- prever, com uma certa taxa de erro, seus desempenhos, o que facilita o uso mais inteligente dos métodos sem precisar utilizar muito esforço computacional devido a testes.
- reforçar ou contrariar as conclusões já estabelecidas dos métodos, uma vez que a maioria delas são baseadas em testes experimentais.

1.2 Objetivos

O objetivo geral deste trabalho é analisar e comparar métodos multirrótulos distintos e desenvolver um novo algoritmo de classificação multirrótulo. O Objetivo geral pode ser detalhado nos seguintes objetivos específicos:

- Descobrir como medir e explorar correlação entre rótulos;
- Análise crítica dos métodos multirrótulos;
- Elaboração de um algoritmo de um novo método multirrótulo.

1.3 Estrutura do Trabalho

O restante do trabalho está organizado da seguinte forma:

- O capítulo 2 apresenta os principais conceitos da classificação multirrótulo, bem como os métodos usados para avaliação de desempenho de classificadores multirrótulo.

- O capítulo 3 apresenta a definição de diferentes métodos de classificação multirrótulo usados neste trabalho, bem como suas complexidades algorítma.
- O capítulo 4 apresenta a definição de um novo método de classificação multirrótulo proposto neste trabalho.
- O capítulo 5 começa apresentando as configurações experimentais escolhidos para realização de testes e termina apresentando os resultados e a sua análise detalhada.
- No capítulo 6 são apresentados as conclusões finais sobre o desempenho dos métodos e sobre a análise do capítulo anterior.

2 Classificação multirrótulo

Problemas de classificação estão situadas na área de aprendizado supervisionado, que por sua vez é uma subárea da mineração de dados. Para (DUNHAM; MING, 2003) a mineração de dados é definido como a descoberta de informações escondidas em um conjunto de dados. Ela surgiu diante do grande crescimento de dados armazenados em arquivos de computadores e do desejo dos usuários desses dados em obter informações mais elaboradas. A mineração de dados tem por objetivo satisfazer o desejo desses usuários ao desenvolver técnicas capazes de explicitar informações valiosas, antes escondidas ao usuários diante de uma alta quantidade de dados. Uma de suas subáreas é a aprendizado supervisionado. Nela, segundo (MOHRI; ROSTAMIZADEH; TALWALKAR, 2012), os dados pelos quais deve-se extrair as informações são divididos em amostras e cada amostra está associado a uma variável especial, de valor conhecido, e a técnica deve predizer o valor dessa variável especial para novas amostras cujo valor da variável especial associada a elas é desconhecido. Normalmente, em aprendizado supervisionado os dados são objetos de um domínio específico e cada um dos objetos é descrito por um conjunto fixo de atributos (REZENDE, 2003). Esses objetos são usualmente chamados de instâncias ou exemplos do domínio do problema. Um atributo é uma descrição de uma característica da instância.

Em problemas de classificação unirrótulo, a variável especial associada a cada instância é discreta e é chamada de classe ou rótulo. A técnica que prediz as classes é também chamada de classificador. Quando existem apenas duas classes, o problema é chamado de classificação binária. Na classificação multirrótulo, cada instância pode assumir um ou mais rótulos, e a técnica que prediz os rótulos de uma instância é chamada de classificador multirrótulo. Por exemplo, uma instância de filme pode ser rotulado como sendo de romance e comédia, e não exclusivamente de romance ou comédia.

Assim a classificação é a tarefa cujo objetivo é encontrar um classificador capaz de predizer o rótulo ou os rótulos de uma instância corretamente. Para completar essa tarefa, o classificador deve usar dados de entrada que são exemplos de treino cujos rótulos são conhecidos afim de reconhecer e aprender padrões neles.

2.1 Enunciado do problema

Em um problema de classificação multirrótulo, seja X o espaço de características tal que $X \subseteq \mathbb{R}^n$ e $L = \{l_1, l_2, l_3, ...l_r\}$ o conjunto dos r rótulos possíveis do problema, uma instância é definida como sendo uma dupla de vetores (x', y') tal que $x' \in X$ e y' é um vetor binário $y' = (y'_1, y'_2, ..., y'_r)$ de tal forma que $y'_i = 1$ indica a presença do rótulo l_i na instância. Assim, o espaço de rótulos possíveis para uma instância qualquer é definido como $Y = \{0,1\}^r$.

Seja f,$f : X \to Y$, a função que mapeia qualquer x,$x \in X$ a seus rótulos reais. A tarefa do problema de classificação multirrótulo é encontrar a função f a partir de uma base de treino $D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}, x_i \in X, y_i \in Y$. Uma vez que é muito difícil encontrar f, ela é aproximada, resultando em $\hat{f}$. Com isso, a tarefa do problema de classificação se torna em aproximar ao máximo $\hat{f}$ de f. Formalmente, a aproximação é medida por uma métrica de qualidade e o objetivo se torna em minimizar a aproximação.

Note que em um problema de classificação unirrótulo todas as instâncias da forma (x', y') tem como y' um vetor binário de rótulos onde apenas uma posição tem valor 1. Assim, podemos ver o problema classificação unirrótulo como um caso específico do problema de classificação multirrótulo. Outro ponto importante a notar é a grande diferença da complexidade da classificação unirrótulo para a multirrótulo. Enquanto que na classificação unirrótulo o número de possíveis rotulações que uma instância desconhecida pode ter é r, linear em relação ao número de rótulos, na multirrótulo o número cresce exponencialmente, a saber, 2^r. Assim construir um classificador multirrótulo é mais complexo que um classificador unirrótulo.

Alguns autores vêem os classificadores unirrótulo e os multirrótulo como uma função de probabilidade p (READ; MARTINO; LUENGO, 2012), (DEMBCZYNSKI; CHENG; HÜLLERMEIER, 2010). No caso de classificadores unirrótulo, um classificador é uma função de probabilidade $p(y|x)$, onde $y \in L$ e $x \in X$, que estima a probabilidade da instância que tem o vetor de características x, ter o rótulo y. Já no caso de classificadores multirrótulo, a função de probabilidade $p(y|x)$, onde $y \subseteq L$ e $x \in X$, estima a probabilidade da instância que tem x, ter todos os rótulos em y. Dessa forma, para um x,$x \in X$ qualquer, a função $\hat{f}$ pode ser obtida pela equação 2.1:

$$\hat{f} = \arg\max_{y^*} p(y|x) \tag{2.1}$$

Apesar da função $\hat{f}$ ser melhor aproximada de f quando obedece a equação 2.1, muitos

métodos multirrótulos não a obedecem por ser muito custoso de estimá-la, uma vez que deve-se procurar y^* dentre 2^r possíveis combinações.

2.2 Avaliação de Desempenho

A avaliação de desempenho de classificadores, tanto multirrótulo quanto unirrótulo, é comumente feito por meio de testes nas amostras coletadas do problema. Isso é feito corretamente com a ajuda de métodos de reamostragem fundamentados pela ciência estatística, descritas na seção 2.2.2.

A avaliação de desempenho dos classificadores multirrótulos se difere da unirrótulo principalmente na quantificação da qualidade de predição. Enquanto que na classificação unirrótulo existe somente uma classificação correta dentre apenas r possíveis classificações, na classificação multirrótulo podem existir mais de uma combinação, dentre as 2^r possíveis, que estejam corretas ou parcialmente corretas. Para isso, são definidas várias métricas multirrótulo na seção 2.2.1, cada uma capturando um aspecto diferente do desempenho do classificador.

2.2.1 Métricas

Seja $P = (p_1, p_2, ..., p_n), p_i \subseteq L$ um vetor de predições de rótulos produzido pela classificação das n instâncias de rótulos $(r_1, r_2, ..., r_n), r_i \subseteq L$ respectivamente. Note que aqui as predições p_i e os rótulos r_i estão representados na forma de conjunto de rótulos, e não na forma de vetor binário. As métricas multirrótulo propostas servem para quantificar a qualidade de P e úteis para resumi-lo a um único valor escalar entre 0 e 1. Abaixo estão algumas métricas definidas por (ZHANG; ZHOU, 2013):

Hamming Loss

$$hloss(P) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|L|}|p_i \triangle r_i| \tag{2.2}$$

O símbolo $\triangle$ é definido como a diferença simétrica entre dois conjuntos, por exemplo, para quaisquer A e B, $A \triangle B = (A \cup B) - (A \cap B)$. O *Hamming Loss* significa a proporção de rótulos preditos mal classificados. Por rótulo predito mal classificado entende-se por classificação do rótulo que não existe na instância real ou a ausência da classificação do rótulo que existe na instância real.

Por rótulo predito mal classificado entende-se que classificou um rótulo que não existia ou deixou de classificar um rótulo relevante. Note que quanto menor o seu valor, melhor é a qualidade de predição, sendo que 0 é a qualidade perfeita e 1 a mais imperfeita possível.

Em (DEMBCZYNSKI; CHENG; HÜLLERMEIER, 2010) é mostrado que para que um classificador minimize o valor dessa métrica, basta minimizar o erro (mal classificação) para cada rótulo individualmente. Assim, para minimizar essa métrica não é necessário levar em consideração a correlação entre rótulos. Dessa forma considerar os rótulos de forma independente é o suficiente para minimizá-lo, apesar de que um método multirrótulo pode usar a correlação entre rótulos para ajudar a minimizá-lo, uma vez que a tarefa de classificação, mesma que de forma independente, é difícil.

Subset Accuracy

$$subsetAcc(P) = \frac{1}{n}\sum_{i=1}^{n} [\![p_i = r_i]\!] \tag{2.3}$$

O *Subset Accuracy* avalia a proporção de instâncias corretamente classificados. Se $|p_i| \neq |r_i|$, então a instância é dita incorretamente classificada. Aqui, nessa métrica, entende-se por instância corretamente classificado quando o conjunto de rótulos preditos é idêntico ao conjunto de rótulos reais. É uma métrica rígida cujo valor ideal é 1 enquanto que o menor valor possível é 0.

Em (DEMBCZYNSKI; CHENG; HÜLLERMEIER, 2010) é mostrado que para maximizar o valor dessa métrica, é necessário levar em consideração a dependência entre rótulos. É por isso que ele é considerado nesse trabalho uma métrica que exige que o classificador multirrótulo explore a correlação entre rótulos.

Example Based Accuracy

$$exampleAcc(P) = \frac{1}{n}\sum_{i=1}^{n} \frac{|p_i \cap r_i|}{|p_i \cup r_i|} \tag{2.4}$$

Note que para essa métrica o valor de melhor desempenho é 1 e o de pior desempenho é 0.

2.2.2 Método de Reamostragem

Um método de reamostragem é um modelo ou processo de avaliação para estimar valores estatísticos (no nosso caso, as métricas definidas na seção 2.2.1) usando apenas subconjuntos dos dados disponíveis (YU, 2003). O método de reamostragem (ou modelo de avaliação) é diferente da métrica de avaliação, pois define como o classificador deve ser avaliado, enquanto que a métrica mede o desempenho, dando um valor para ele.

Na área de Aprendizado Supervisionado, um dos métodos de reamostragem mais usado é a Validação Cruzada. Para um k pré-definido maior que 1, a Validação Cruzada consiste em dividir o conjunto de dados $\boldsymbol{D}$ em k subconjuntos disjuntos de mesma cardinalidade ,$\{s_1, s_2, ..., s_k\}$ tal que $s_1 \cup s_2 \cup s_3 ... s_{k-1} \cup s_k = \boldsymbol{D}$, e realizar k testes, enumerados de 1 a k. Cada teste i, para $1 \leq i \leq k$, separa um desses subconjuntos para formar a base de dados de teste enquanto os restantes formam a base de dados de treino. Em seguida, o modelo de classificação é treinado sobre a base de treino e testado sobre todas as instâncias da base de teste. No final, juntando os k testes, temos uma classificação (predição) para cada instância de $\boldsymbol{D}$. A partir daí aplica-se as métricas de avaliação, como por exemplo o *Hamming Loss*.

3 Métodos Multirrótulos

3.1 Relevância Binária - BR

O método da Relevância Binária, conhecido como *Binary Relevance* (TSOUMAKAS; KATAKIS; VLAHAVAS, 2010), é composto de r classificadores binários $c_1, c_2, ..., c_r$. Cada classificador c_i é associado ao rótulo i e treinado com o único objetivo de resolver um problema de classificação binária onde as instâncias que tem o rótulo r_i são consideradas para o classificador c_i como positivas e as demais instâncias como negativas. Após todos os classificadores terem sido treinados, quando uma instância de rótulo desconhecido é apresentado aos classificadores, todos aqueles que produzirem uma classe positiva terão sua classe associada à nova instância. O método de classificação de relevância binária é uma estratégia de transformação do problema, que decompõe o problema de classificação multirrótulo em diversos problemas de classificação binária unirrótulo, um para cada um dos rótulos do problema.

A figura 3.1 ilustra um exemplo da transformação que o BR realiza em um problema multirrótulo de rótulos A, B e C e 6 instâncias. Nele vemos que o BR transforma a base de treino em 3 novas bases de dados, um para cada classificador binário.

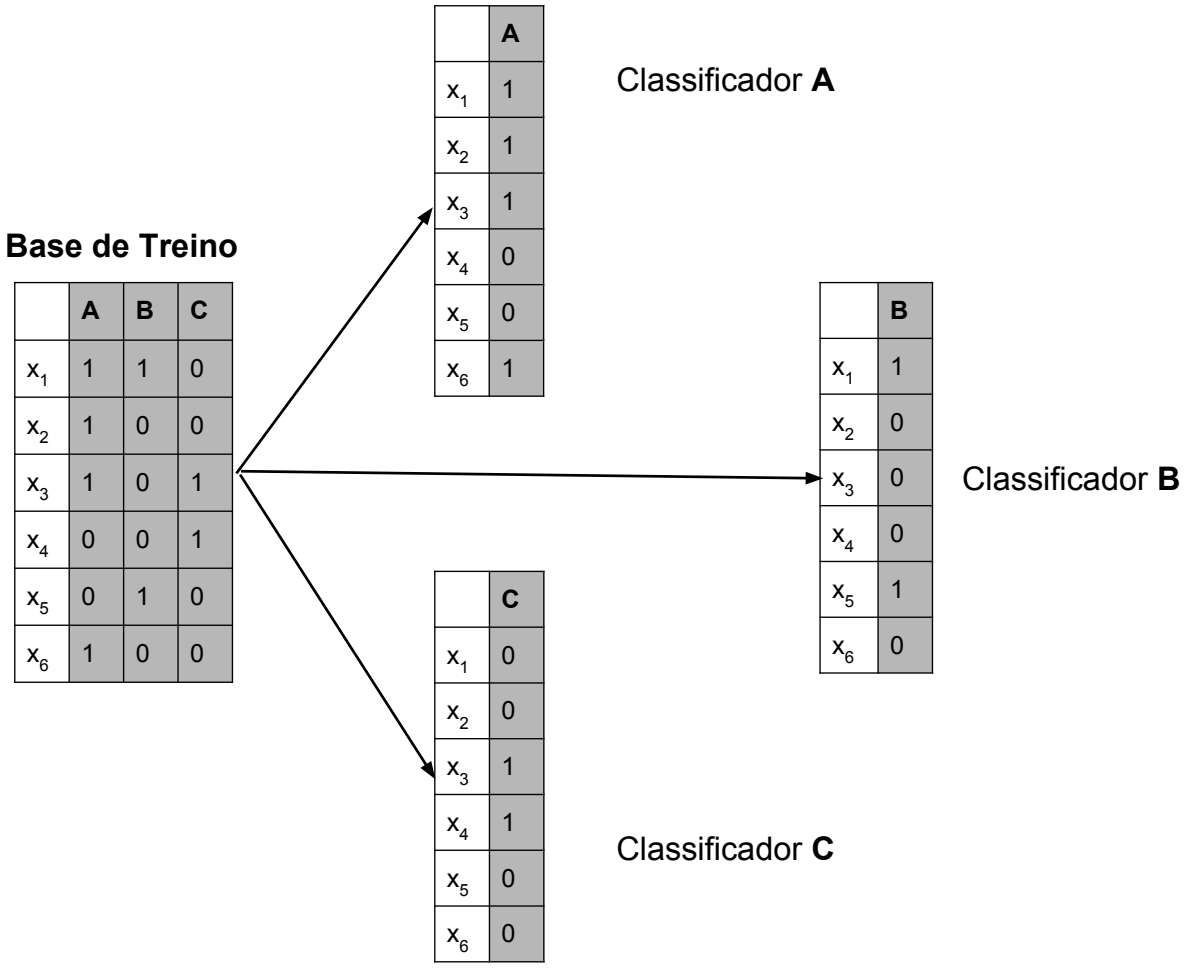


Figura 3.1: Exemplo da transformação realizada pelo método *BR*

3.2 Classifier Chain

A idéia básica desse algoritmo é semelhante ao BR: realiza a transformação do problema multirrótulo decompondo-o em diversos problemas de classificação binária unirrótulo, um para cada um dos rótulos do problema. Ele é também composto de r classificadores binários $c_1, c_2, ..., c_r$ e cada um é associado a um único rótulo distinto. A diferença do *Classifier Chain* para o BR está em que os classificadores binários estão organizados em uma cadeia de tal forma que o classificador c_i é contruído com base nos rótulos ou predições dos classificadores anteriores $(c_{i-1}, c_{i-2}, ..., c_1)$ (READ et al., 2009). O classificador c_i não está necessariamente associado ao rótulo r_i, ele pode estar associado a qualquer um dos rótulos. Essa associação é feita de forma aleatória ou pré-definida por parâmetro do algoritmo.

Na fase de treinamento do método o espaço de características de cada classificador c_i é extendido com os valores dos $i-1$ rótulos reais anteriores da cadeia. Veja um exemplo ilustrado na figura 3.2 onde o método é treinado sobre uma base de treino de três rótulos (A, B, C) e seis instâncias. Note que a base de treino do classificador binário B tem como característica adicional o rótulo A.

Na fase de predição do método a classificação ocorre de forma sequencial, na ordem em que a cadeia foi definida. O classificador c_1 inicia o processo de classificação realizando a estimativa do rótulo associado da instância teste. A partir daí o classificador c_i realiza a predição da instância teste assim que a predição do classificador c_{i-1} estiver disponível. O classificador c_i agrega a predição do classificador c_{i-1}, que é um valor binário (0 ou 1), a instância de teste. A figura 3.2 ilustra bem o processo de predição na qual o vetor de características da instância teste vai crescendo com adição de cada estimativa de rótulo.

Dessa forma, o *Classifier Chain* considera a dependência entre os rótulos, pois a predição de um de seus classificadores binários afeta diretamente na predição dos classificadores binários seguintes.

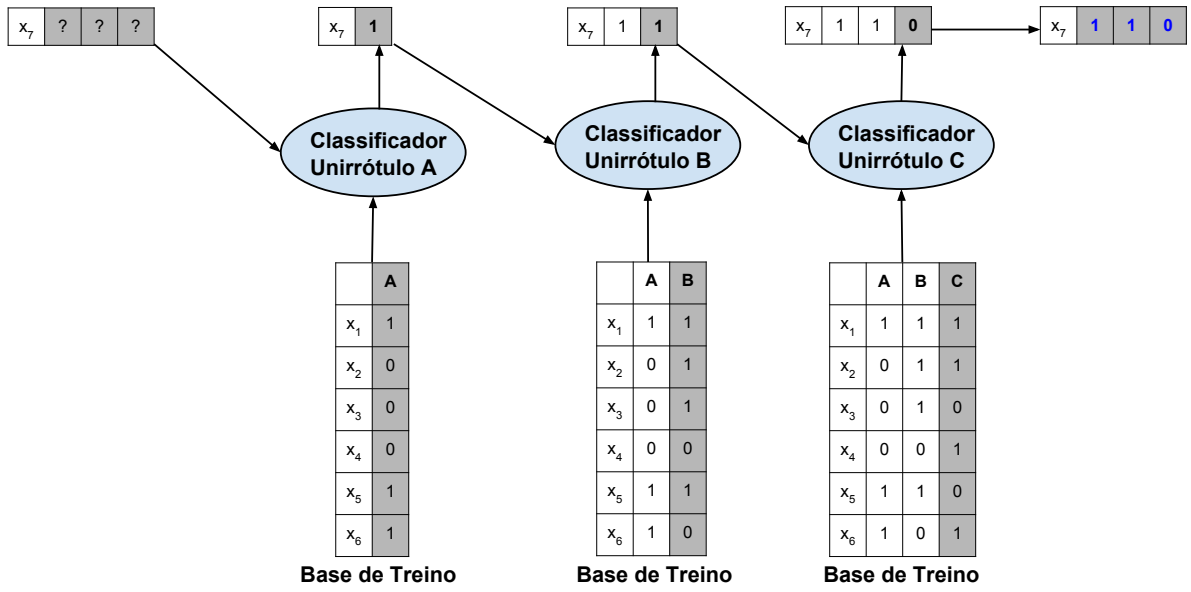


Figura 3.2: Ilustração de um exemplo da fase de treinamento e de predição do método Classifier Chain.

3.3 Ensemble of Classifier Chain

O *Ensemble of Classifier Chain* é composto de k *Classifiers Chain* distintos (READ et al., 2009), para um k pré-definido. Para cada um dos k *Classifiers Chain* é definido uma ordem aleatória da cadeia e cada um é treinado sobre um amostra aleatória da base de dados de treino. Na fase de predição, a instância é submetida a todos os *Classifiers Chain*, resultando em k predições individuais para uma mesma instância. A predição final é feita combinando as k predições individuais, sendo feito por voto majoritário, ou seja, um rótulo y estará na predição final se y estiver em pelo menos $\lceil \frac{k}{2} \rceil$ das predições individuais. O motivo para que cada *Classifier Chain* do *Ensemble* ser diferente é que ao combinar um classificador **c** com outros classificadores diferentes, espera-se que nas instâncias em que **c** classificar incorretamente, a maior parte dos outros classificadores classifiquem corretamente.

3.4 Relevância Binária Dependente - DBR

Este método é proposto por (MONTANES et al., 2014) e é baseado no método BR e a grande diferença entre ambos está no fato de que o DBR considera dependência entre os r rótulos.

O DBR é composto de dois classificadores multirrótulo, c_0 e c_1 , cada um composto

de r classificadores binários. O classificador multirrótulo c_0 é exatamente o método BR. Os r classificadores binários $c_1^1, c_1^2 ..., c_1^r$ que compoêm c_1 trabalham em um novo espaço de características $X^{new} = X \times \{0,1\}^{r-1}$. Esse novo espaço é a extensão do antigo com a adição de $r-1$ rótulos. Digamos que (x,y) seja uma instância do espaço original $X \times Y$ onde $x \in X$ e $y \in \{0,1\}^r$, então cada instância do classificador binário c_1^i tem $|x| + r - 1$ características e é definido como sendo $(x, y_1, ..., y_{i-1}, y_{i+1}, ..., y_r)$.

3.4.1 Fase de Treinamento

Seja $D = \{((x_i), y_i) | i = 1, ..., n\}$ a base de dados de treino composta de n instâncias, onde $x_i \in X$ e $y_i \in Y$, x_i é o vetor de características de cada instância e y_i o vetor binário de rótulos de cada instância. O método primeiro treina o c_0 no espaço de características original conforme o treinamento do próprio BR mostrado na seção 3.1. Depois, treina-se c_1 em uma nova base de dados D' que é construída a partir de D ao adicionar os rótulos de cada exemplo como características. Assim, D' é composta pelos exemplos $\{((x_i, y_i), y_i) | i = 1, ..., n\}$ e cada classificador binário c_1^j de c_1 é induzido na base de dados $D'_j = \{(x_i, y_{i,1}, ..., y_{i,j-1}, y_{i,j+1}, ..., y_{i,r}), y_{i,j} | i = 1, ..., n\}$. Note que a característica representando o j-ésimo rótulo é removido da base de dados. Dessa forma, ao invés de cada classificador binário ser uma função que depende apenas do vetor de características, como o BR, o método é capaz de detectar dependência entre os rótulos pelo fato de cada classificador binário c_1^j ser definido por uma função que depende adicionalmente dos valores dos rótulos sendo preditos.

3.4.2 Fase de Predição

Como no caso do Classifier Chain, os rótulos reais y, que são usados como características adicionais em cada instância de treino, estão disponíveis apenas durante a fase de treinamento. Com isso, para tornar possível a classificação por c_1, o DBR usa o classificador c_0 com a finalidade de estimar os rótulos, o que resulta na predição $c_0(x) = \hat{y} = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_r)$, que servirá como parte da instância a ser classificada por c_1, onde antes era o lugar de y. A partir daí, c_1 classifica o vetor de características $(x, \hat{y})$ de uma forma bem similar ao BR: cada classificador binário c_1^i do método é responsável pela predição de um único rótulo da instância cujo vetor de características é $(x, \hat{y}_1, ..., \hat{y}_{j-1}, \hat{y}_{j+1}, ..., \hat{y}_r)$.

3.5 Monte Carlo Classifier Chain

O método *Classifier Chain with Monte Carlo Optimization* (MCC) é introduzido por (READ; MARTINO; LUENGO, 2012) para melhorar a qualidade de predição do *Classifier Chain*. O método usa uma heurística gulosa para melhorar a fase de predição do *Classifier Chain*. O método parte do pressuposto de que o *Classifier Chain* geralmente não encontra o conjunto de rótulos $y^*, y^* \in \{0,1\}^r$ cuja probabilidade de a instância a tem seja máxima, ou seja, não segue corretamente a equação 2.1. Para encontrar o y^* de maior probabilidade, a princípio deve-se procurá-lo em todas as possíveis combinações de $\{0,1\}^r$, o que torna-se inviável para um valor de $r > 10$. O MCC tenta encontrar esse y^* sem testar todas as possíveis 2^r combinações do espaço de busca. Ele usa de um algoritmo de otimização, chamado *Monte Carlo* (DICKMAN; GILMAN, 1989), para testar apenas algumas dessas combinações.

4 Recursive Dependent Binary Relevance - RDBR

A proposta de Recursive Dependent Binary Relevance (RDBR) é fundamentada no método multirrótulo *dependent binary relevance* (DBR) (MONTANES et al., 2014), que é explicado na seção 3.4. Assim como o DBR e o CC, o RDBR é um método baseado na transformação do problema que dividem o problema multirrótulo em vários problemas classificação binária. Todos eles exploram a correlação entre os rótulos por meio da adição de características especiais que representam os rótulos reais ou estimativas dos rótulos reais ao espaço de características original. Mas, diferentemente dos outros, o RDBR adiciona uma inteligência no uso dessas características especiais na fase de predição do método. A forma de como isso é feito, bem como o funcionamento completo do algoritmo de classificação e a fundamentação teórica do RDBR são detalhados na seção 4.1. A seção 4.2 analisa o funcionamento e o desempenho do RDBR de forma empírica.

4.1 Algoritmo de Recursive Dependent Binary Relevance - RDBR

Como foi dito anteriormente, o Recursive Dependent Binary Relevance é baseado no DBR. Ambos dependem da hipótese de que as estimativas dos rótulos em Y por um classificador multirrótulo c_0 são boas características para aprimorar as estimativas dos mesmos rótulos por um novo classificador c_1 e que quanto melhor forem as estimativas dos rótulos por c_0, melhores são as de c_1. E ainda ambos usam o BR como classificador multirrótulo base, que servirá para realizar as primeiras estimativas dos rótulos.

No entanto, o RDBR, ao invés de usar apenas o classificador base c_0 como função para contruir as características adicionais que o c_1 usa, como o DBR, ele também usa o próprio c_1 para essa finalidade, ou seja, há uma atualização das estimativas das características pelo próprio classificador que as usam. A idéia é que cada vez que c_1 as atualiza, melhores

ficam suas estimativas uma vez que ele será baseado em estimativas melhores de rótulos do que anteriormente. O funcionamento do algoritmo é detalhado nas subseções 4.1.1 e 4.1.2. Formalmente, a estrutura do RDBR é organizado da seguinte forma:

- Assim como o DBR, é composto de um BR e um classificador multirrótulo, c_0 e c_1, cada um composto de r classificadores binários.
- O c_0 trabalha dentro do espaço de características original do problema, de nome X, e c_1 trabalha dentro de um novo espaço de características do problema, de nome X_e e definido como $X_e = X \times \{0,1\}^r$. Assim, c_0 e c_1 são representados pelas seguintes funções:

$$\begin{aligned} c_0 &: X \to \{0,1\}^r \\ c_1 &: X_e \to \{0,1\}^r \end{aligned} \tag{4.1}$$

- Os r classificadores binários $c_1^1, c_1^2 ..., c_1^r$ que compoêm c_1 não trabalham no mesmo espaço de características, contudo, trabalham com uma dimensão reduzida, em $X \times \{0,1\}^{r-1}$. Digamos que (x,y) seja uma instância de X_e onde $x \in X$ e $y \in \{0,1\}^r$, então cada instância do classificador binário c_1^i tem $|x| + r - 1$ características e é definido como sendo $(x, y_1, ..., y_{i-1}, y_{i+1}, ..., y_r)$.

As seções seguintes explicam o funcionamento da estrutura apresentada bem como formalizam e detalham tanto a fase de treinamento quanto a fase de predição do algoritmo. É importante observar que a fase de treinamento do RDBR é exatamente o mesmo do que o DBR. A diferença de ambos os métodos se dá na fase de predição, descrita na seção 4.1.2.

4.1.1 Fase de Treinamento

A fase de treinamento do RDBR é exatamente igual ao DBR.

Formalmente, o treinamento de Recursive Dependent Binary Relevance funciona da seguinte forma. Seja $D = \{((x_i), y_i) | i = 1, ..., n\}$ a base de dados de treino composta de n instâncias, onde $x_i \in X$ e $y_i \in Y$, x_i é o vetor de características de cada instância e y_i o vetor binário de rótulos de cada instância. O algoritmo primeiro treina o classificador multirrótulo c_0 no espaço de características original conforme o treinamento do próprio BR mostrado na seção 3.1. Depois, treina-se c_1 em uma nova base de dados D' que é construída a partir de D adicionando os rótulos de cada exemplo como características.

Assim, D' é composta pelos exemplos $\{((x_i, y_i), y_i) | i = 1, ..., n\}$ e cada classificador binário c_1^j de c_1 é induzido na base de dados $D'_j = \{(x_i, y_{i,1}, ..., y_{i,j-1}, y_{i,j+1}, ..., y_{i,r}), y_{i,j} | i = 1, ..., n\}$. Note que a característica representando o j-ésimo rótulo é removido da base de dados. Dessa forma, ao invés de cada classificador binário ser uma função que depende apenas do vetor de características, como o BR, o método é capaz de detectar dependência entre os rótulos pelo fato de cada classificador binário c_1^j ser definido por uma função que depende adicionalmente dos valores dos rótulos sendo preditos.

4.1.2 Fase de Predição

O funcionamento do RDBR distingue-se do DBR apenas na fase de predição. Dado o vetor de características x de uma instância onde $x \in X$ e seu conjunto de rótulos reais $y, y \in \{0,1\}^r$, queremos que a função C representando o classificador multirrótulo RDBR, onde $C : X \to Y$, retorne y quando o submetemos x, ou seja, $C(x) = y$.

Como no caso do DBR e do Classifier Chain, os rótulos reais y, que são usado como características especiais, estão disponíveis apenas durante a fase de treinamento. Dessa forma, para tornar possível a classificação por c_1, usou-se o classificador multirrótulo c_0 para estimar os rótulos, resultando em $c_0(x) = \hat{y}^0 = (\hat{y}_1^0, \hat{y}_2^0, ..., \hat{y}_r^0)$, que servirá como parte da instância de c_1 no lugar de y. A partir daí, c_1 classifica o vetor de características $(x, \hat{y}^0)$ de uma forma bem similar ao BR: cada classificador binário c_1^i do método é responsável pela predição de um único rótulo da instância cujo vetor de características é $(x, \hat{y}_1^0, ..., \hat{y}_{j-1}^0, \hat{y}_{j+1}^0, ..., \hat{y}_r^0)$. Esse procedimento é o realizado pelo DBR e é ilustrado na figura 4.1. Nela vemos quais estimativas de rótulos são utilizadas como características adicionais para classificação final de uma instância.

Assim que c_1 classifica a instância $(x, \hat{y}^0)$, gerando portanto a estimativa de rótulos $\hat{y}^1 = c_1(x, \hat{y}^0)$, $\hat{y}^1$ é usado para atualizar as características da instância x, tomando assim o lugar de $\hat{y}^0$. Esse processo de atualização das características é iterativo e é repetido k vezes, onde k é determinado por um valor máximo de iterações, definido a priori, ou quando é detectado a convergência. A convergência é alcançada quando a estimativa de rótulos não muda, independente do número de iterações. Com k iterações, tem-se k estimativas de rótulos $\hat{y}^1, \hat{y}^2, ..., \hat{y}^k$, dentre as quais o último $(\hat{y}^k)$ é a classificação final do método $C(x) = \hat{y}^k$.

Dessa forma, podemos concluir que Recursive Dependent Binary Relevance é um método recursivo de tal forma que para $k = 1$, $C(x) = c_1(x, c_0(x))$, para $k = 2$, $C(x) = c_1(x, c_1(x, c_0(x)))$, para $k = 3$, $C(x) = c_1(x, c_1(x, c_1(x, c_0(x))))$ e assim por diante. Note que

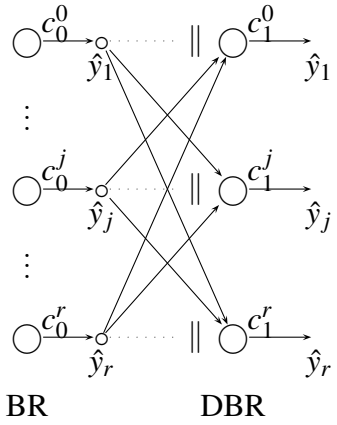


Figura 4.1: Arquitetura do classificador *Dependent Binary Relevance* (DBR). Na primeira camada (a esquerda), os classificadores binários do BR proveem cada um dos rótulos individualmente. A próxima camada provê a estimativa final dos rótulos.

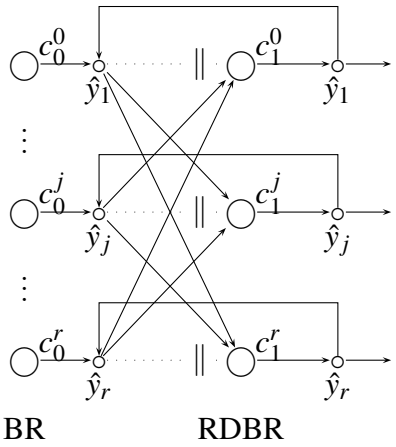


Figura 4.2: Arquitetura do *Recursive Dependent Binary Relevance* (RDBR). Na primeira camada (a esquerda), os classificadores binários do BR proveem estimativas de cada um dos rótulos individualmente. A próxima camada provê as estimativas obtidas pelo DBR as quais são usados recursivamente ao realimentar o DBR.

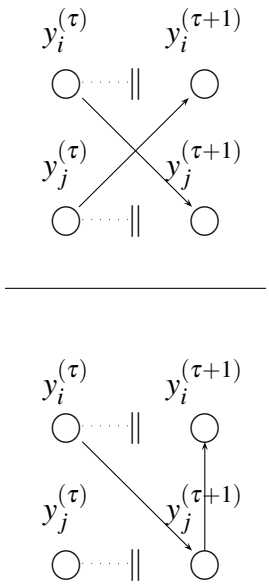


Figura 4.3: Estrutura do RDBR com atualização estática (imagem acima) e dinâmica (imagem abaixo) dos rótulos. Na atualização dinâmica, a estimativa dos rótulos $\hat{y}_i{}^{(\tau+1)}$ é baseado nas estimativas dos rótulos anteriores (da iteração (τ)) e nas da iteração atual $((\tau+1))$, se disponíveis.

para $k=0$, o RDBR é exatamente o BR, $C(x)=c_0(x)$. Aplicando esse processo recursivo, espera-se que a cada recursão i a estimativa dos rótulos $\hat{y}^i$ seja melhor do que seu antecessor $\hat{y}^{i-1}$. Teoricamente, essa afirmação se mantém se supormos que a estimativa $\hat{y}^1$ é melhor do que a $\hat{y}^0$, o que é razoável uma vez que o classificador c_0, que é um BR, obtêm seu resultado usando apenas estimativas marginais dos rótulos, enquanto que c_1 explora a correlação dos rótulos ao usá-los como características, obtendo assim estimativas baseadas na probabilidade condicional. Com essa suposição teríamos que $\hat{y}^i$ seria melhor do que $\hat{y}^{i-1}$, pois $\hat{y}^{i-1}$ se aproxima mais da distribuição real dos rótulos do que $\hat{y}^{i-2}$. Assim, quando c_1 estimar $\hat{y}^i$ usando $\hat{y}^{i-1}$ estaria baseado em uma distribuição mais próxima daquela em que foi treinado do que usando $\hat{y}^{i-2}$. Lembrando que c_1 foi treinado usando apenas rótulos assumidamente corretos. Olhando por todo o procedimento descrito, o RDBR pode ser simplesmente visto como uma generalização do BR e do DBR que insere uma inteligência adicional a aplicação e uso do classificador c_1 de DBR, afim de que ele seja melhor aproveitado. A figura 4.2 ilustra bem o funcionamento do RDBR. Nela vemos quais estimativas de rótulos são utilizadas como características adicionais para próxima estimativa de rótulos.

Adicionalmente, o RDBR adota uma técnica extra, inspirada no Classifier Chain que consiste em, para cada classificador binário c_1^j, atualizar a característica $\hat{y}_j$ imediatamente após a sua classificação. Dessa forma, os classificadores binários seguintes, $c_1^{j+1}, c_1^{j+2}, ..., c_1^r$,

classificarão suas instâncias baseados em estimativas de rótulos mais atuais, possivelmente melhores. Isso é ilustrado na figura 4.3 e é chamado de atualização dinâmica dos rótulos.

4.2 Análise

Nessa seção o método RDBR é posto em prova. Com objetivo de analisar o método, implementou-se o algoritmo usando a linguagem de programação Java e no *Weka* (HALL et al., 2009), que é uma biblioteca que integra técnicas de reconhecimento de padrões. A principal hipótese em que o RDBR é baseado será testado nessa seção com o intuito de validar o método. Com a finalidade de tornar os testes mais objetivos, a hipótese é melhor formalizada assim:

- Dados uma métrica M, uma base de Teste $D = \{x_1, x_2, ..., x_n\}$, um DBR induzido composto pelos classificadores multirrótulos c_0 e c_1 e dois vetores de predições de r rótulos:

$$\begin{aligned} p &= (p_1, p_2, ..., p_n) : p_i \in \{0,1\}^r | 1 \leq i \leq n \\ b &= (b_1, b_2, ..., b_n) : b_i \in \{0,1\}^r | 1 \leq i \leq n \end{aligned} \tag{4.2}$$

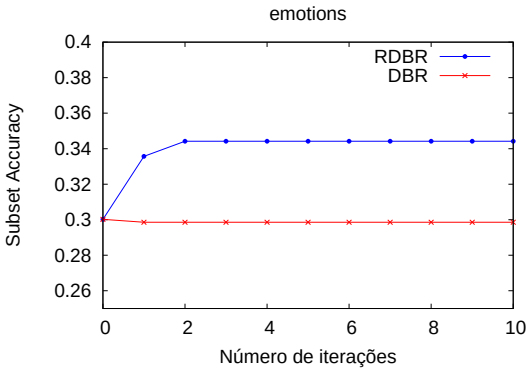
  tal que $M(p_2) \geq M(p_1)$, então:

$$M((c_1(x_i, p_1)|1 \leq i \leq n)) \leq M((c_1(x_i, p_2)|1 \leq i \leq n)) \tag{4.3}$$

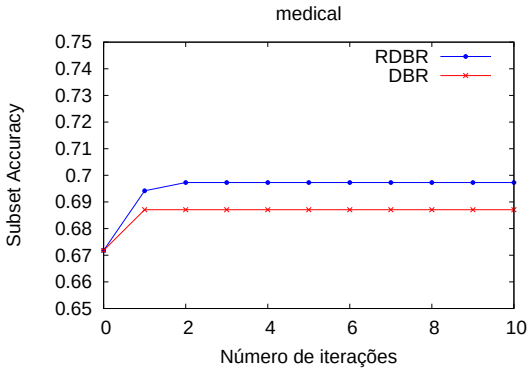
Resumidamente, a hipótese é que erros de predições pelo classificador c_0 do DBR afetam negativamente a classificação do classificador c_1. A comprovação dessa hipótese é feita da seguinte forma. Experimentos com o RDBR são realizados usando 7 bases de dados de domínio públicos. Cada experimento consiste em medir o valor da métrica *Subset Accuracy* quando o método é submetido a validação cruzada de 10 *folds*. O experimento é repetido com o número máximo de iterações do RDBR variando de 0 a 10 (Note que para o valor 0, o RDBR se torna exatamente o BR). Ao variar esse parâmetro, espera-se que o método obtenha desempenho melhor para os valores mais altos. De fato, é o que ocorre na maioria dos casos, apesar de que o método converge rapidamente em relação ao número de iterações. Os gráficos da figura 4.4 mostram o que ocorre em 5 dos 7 casos testados: o método tem seu desempenho melhorado até o valor máximo de iterações chegar a 2, depois disso o método não tem seu desempenho alterado. Portanto 2 foi o valor máximo de iterações necessárias para o método convergir nesses casos. Nos outros dois casos o método convergiu com apenas uma iteração ou piorou com duas ou mais

iterações. Veja os dois gráficos dos dois casos na figura 4.5. Vale ressaltar que em 6 dos 7 casos, o método RDBR conseguiu um desempenho melhor do que o DBR e em apenas um dos casos alcançou o mesmo desempenho do DBR.

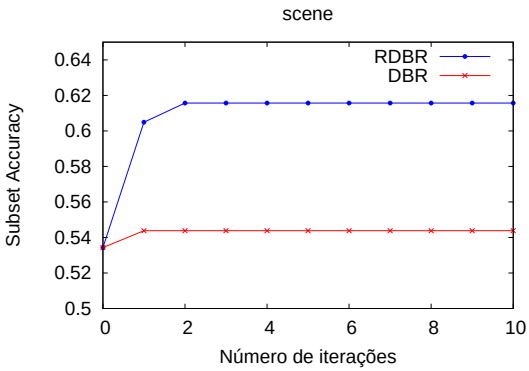
Seja r o número de rótulos, n o número de instâncias de treino e m o número de atributos da base original, na fase de treinamento do RDBR, o número de base de dados utilizadas são $2r$, cada uma contendo n instâncias. Em metade delas, as instâncias contidas tem m atributos e na outra metade $m+r$ atributos. Já na fase de predição do algoritmo, no pior caso, o algoritmo usa $(k+1)r$ bases de dados de n instâncias onde na primeira base, o número de atributos é igual a m e nas restantes é igual $m+r$. Vale ressaltar que nem todas essas bases de dados precisam ser armazenados explicitamente na mémoria em espaços diferentes, algumas são reutilizadas no processo de predição.



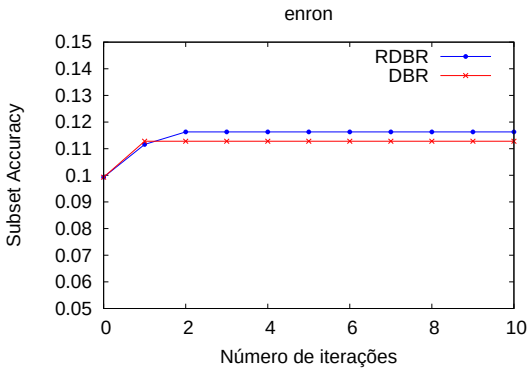
(a) Emotions



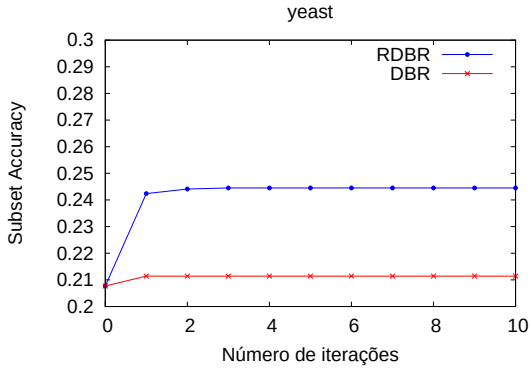
(b) Medical



(c) Scene

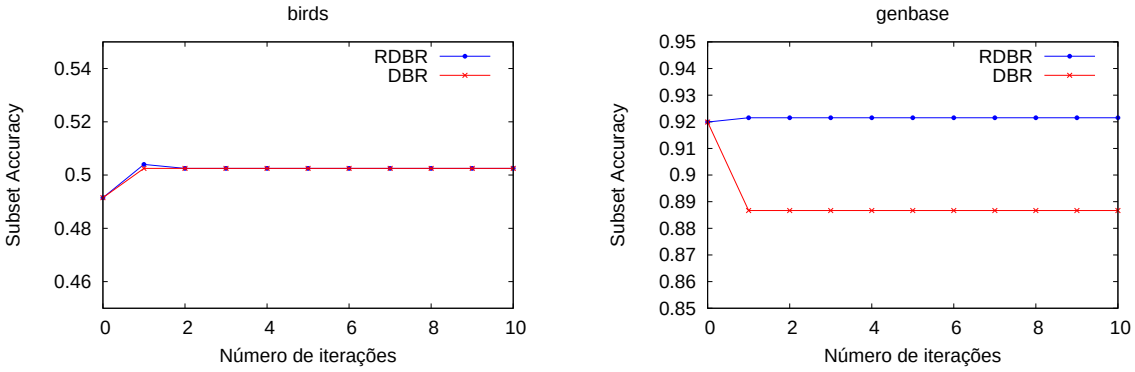


(d) Enron



(e) Yeast

Figura 4.4: Gráficos de análise de desempenho do RDBR.



(a) Birds (b) Genbase

Figura 4.5: Gráficos de análise de desempenho do RDBR nos dois experimentos em que o RDBR não melhorou na sua segunda iteração.

5 Avaliação e Análise Experimental

Neste capítulo é descrito as bases de dados multirrótulo usadas nos experimentos, as medidas de avaliação escolhidas e a escolha dos parâmetros dos classificadores. A comparação e estudo dos métodos é feita considerando a qualidade de predição. A qualidade de predição é estimada pelo método de avaliação por validação cruzada com 10 grupos (*10-fold cross-validation*), descrito na seção 2.2.2, sobre 8 bases de dados, todas apresentadas e descritas na subseção 5.1.

Para quantificar a qualidade de predição, 3 métricas foram utilizadas. As métricas escolhidas, bem como o motivo das escolhas, são listadas:

- *Subset Accuracy*, pois é mostrado que captura bem se o método explora a correlação entre rótulos;
- *Hamming Loss*, pois é bem mais sensível que o Subset Accuracy;
- *Example Based Accuracy*, é o meio termo entre o Hamming Loss e o Subset Accuracy.

As fórmulas para o cálculo de cada uma das métricas são apresentadas na seção 2.2.1. É interessante mostrar os resultados experimentais usando diferentes métricas, pois cada uma captura um aspecto diferente da classificação.

Os experimentos foram realizados utilizando 6 modelos de classificação multirrótulo: BR, DBR, RDBR, CC, ECC e MCC. Os classificadores bases usados, aqueles que são utilizados como classificadores binários para cada um dos métodos multirrótulos citados, foram os seguintes: KNN, SVM, C4.5 e Regressão Logística. Implementações públicas dos classificadores na biblioteca *Weka* (HALL et al., 2009) foram usados para este trabalho. A seguir a descrição de cada um dos classificadores binários.

- **KNN**: O KNN é uma das técnicas do aprendizado supervisionado mais conhecida. O KNN funciona da seguinte forma: Uma instância de características x',$x' \in X$ e de

classe desconhecida é classificado baseado nas classes dos K vizinhos mais próximos, onde mais próximo entende-se por aquela instância (x,y), para $x \in X$ e $y \in Y$, cuja distância euclidiana entre os vetores reais x e x' é mínima. A classe predominante dos K vizinhos mais próximos será a classe atribuída para x' (WU et al., 2008).

- **SVM**: Em um problema de classificação binária, o objetivo do SVM é encontrar o melhor hiperplano, definida pela função de hiperplano $h(x) = 0$ para $x \in X$, que separa as duas classes na base de treino. O "melhor" hiperplano é definido como aquele que simultaneamente minimiza o erro de classificação e maximiza a distância do hiperplano à instância mais próxima dele (WU et al., 2008).

- **C4.5**: O C4.5 é um algoritmo para contruir uma árvore de decisão. Segundo (PENG; CHEN; ZHOU, 2009) uma árvore de decisão é uma árvore cujos nós folhas representam decisões e os nós não-folhas representam uma escolha entre alternativas. Cada nó não-folha é associado a um atributo da base e cada ligação desse nó a seus nós filhos está associado a um valor possível desse atributo. Cada nó folha é associado a uma classe. Quando uma instância de teste é submetida ao classificador, o algoritmo começa no nó raiz e caminha em direção ao nó filho cuja ligação está associado ao mesmo valor do atributo que a instância de teste tem. O algoritmo continua nesse mesmo processo, caminhando sobre os nós e ligações até chegar a um nó folha, onde então a instância de teste é classificado com a classe associada ao nó folha. A quantidade de nós e as associações dos nós aos seus respectivos atributos,valores de atributos ou classes são feitas pelo algoritmo de construção da árvore, que nesse trabalho utilizamos o C4.5.

- **Regressão Logística** Segundo (JAMES et al., 2013) o classificador baseado na Regressão Logística é um modelo probabilístico da estatística. Num problema de classificação binária, esse classificador usa um modelo de regressão não-linear para calcular a probabilidade de um instância qualquer pertencer a classe positiva. O modelo de regressão não-linear usado é a função logística,

$$p(\boldsymbol{y} = 1|\boldsymbol{x}) = \frac{e^{\boldsymbol{w}\cdot\boldsymbol{x}}}{1 + e^{\boldsymbol{w}\cdot\boldsymbol{x}}} \tag{5.1}$$

onde $\boldsymbol{x}$ é o vetor de características, $\boldsymbol{y}$ é a classe positiva e $\boldsymbol{w}$ é um vetor de coeficientes de mesma dimensão que $\boldsymbol{x}$ o qual é estimado durante a fase de treinamento do algoritmo pela máxima verossimilhança (JAMES et al., 2013). Pela equação 5.1 podemos calcular a probabilidade de uma instância de características $\boldsymbol{x}$ pertencer a classe $\boldsymbol{y}$.

A análise experimental dos métodos se encontram divididos em dois estudos. O primeiro estudo consiste em comparar os métodos quando todos usam um mesmo classificador base em específico e o segundo em comparar cada combinação de um modelo de classificação multirrótulo com um classificador base. Ambos estudos são detalhados na seção 5.2.

5.1 Base de dados

As bases de dados são apresentadas na tabela 5.1. Sete das oito bases de dados utilizadas nos experimentos foram obtidas do repositório público de endereço virtual http://mulan.sourceforge.net/datasets.html. A única base de dados não obtida pelo repositório público acima é a nomeada Motorpump, que é uma base de dados privada (MENDEL et al., 2008).

BASE	DOM	EXEMPLOS	DIS	NUM	RÓTULOS	CARD	DENS
Birds	Audio	645	2	258	19	1.014	0.053
Emotions	Música	593	0	72	6	1.869	0.311
Enron	Texto	1702	1001	0	53	3.378	0.064
Genbase	Biologia	662	1186	0	27	1.252	0.046
Medical	Texto	978	1449	0	45	1.245	0.028
Motorpump	Vibração	1372	0	40	9	2.249	0.250
Scene	Imagem	2407	0	294	6	1.074	0.179
Yeast	Biologia	2417	0	103	14	4.237	0.303

Tabela 5.1: Resumo das bases de dados multirrótulos

A tabela 5.1 apresenta algumas estatísticas das bases de dados adquiridas. Nela são apresentadas as seguintes informações de cada base de dados:

- **DOM**: Domínio pertencente;
- **DIS**: Número de atributos discretos;
- **NUM**: Número de atributos numéricos;
- **CARD**: Cardinalidade de rótulos na base de dados, que significa o número médio de rótulos por exemplo;
- **DENS**: Densidade de rótulos na base de dados. Calculado pela divisão da cardinalidade pelo número de possíveis rótulos.

5.2 Resultados Experimentais

Nesta seção é feita uma análise do desempenho dos métodos multirrótulos em cada uma das bases de dados em diferentes métricas. A análise experimental dos métodos se encontram divididos em dois estudos. O primeiro estudo consiste em comparar os métodos quando todos usam um mesmo classificador base em específico e o segundo em comparar cada combinação de um modelo de classificação multirrótulo com um classificador base. No primeiro estudo, que chamaremos de Estudo Específico, para cada um dos 4 classificadores bases, cada método multirrótulo é testado e enumerado de 1 a 6 segundo a ordem de melhor desempenho. O método que obtiver o melhor desempenho, lhe é atribuído o valor 1, o segundo melhor, lhe é atribuído 2 e assim por diante. Essa enumeração é chamada de *ranking* e o valor atribuído de *rank*. No segundo estudo, que chamaremos de Estudo Geral, o *rank* de cada método varia de 1 a 24, uma vez que cada combinação de um modelo de classificação multirrótulo com um classificador base é considerado um método novo. Note que no Estudo Geral o *rank* de um método multirrótulo que tem como o classificador base, por exemplo o KNN, é afetado pelo desempenho do mesmo método porém com outro classificador base. O Estudo Específico tem por principal objetivo evitar que isso aconteça. Adicionalmente, o Estudo Específico tem por objetivo analisar se cada método multirrótulo apresenta desempenhos diferentes para diferentes classificadores, ou seja, se o *ranking* é alterado ao alterar o classificador base. O Estudo Geral tem por objetivo analisar as possíveis combinações de classificadores bases e modelo de classificação multirrótulo sem tornar a comparação e o *ranking* específicos para um subconjunto dessas possíveis combinações. O motivo disso é tomar um ponto de vista em que se desconhece o classificador base que melhor se adapta a cada base a priori. Essa seção é dividida em 2 subseções, um para o Estudo específico e outro para o Estudo Geral, ambos definidos no ínicio do capítulo.

No Estudo Específico os métodos tiveram seus classificadores bases fixados. Ao fixarmos o classificador base, o desempenho irá depender apenas do modelo de classificação multirrótulo de cada um. Em cada subseção e para cada métrica escolhida é apresentado uma tabela contendo os valores da métrica para cada um dos métodos multirrótulo e o ranking dos métodos multirrótulos.

5.2.1 Estudo Específico

Vejamos alguns pontos interessantes nos testes das tabelas 5.2, onde o KNN foi utilizado. Note que o *ranking* dos métodos mudou bastante de métrica para métrica. No caso do BR, o ranking médio caiu de 4.5 no *Subset Accuracy* para 1.75 no *Hamming Loss*, ou seja, de penúltimo colocado no ranking médio para o primeiro colocado. Isso não só ocorre com o KNN, mas também para os outros classificadores bases, cujos resultados se encontram nas tabelas 5.3, 5.4 e 5.5. Note que em todas as tabelas, para a base de dados *Enron*, o valor do *Subset Accuracy* é o mais baixo dentre todas as bases. Isso é esperado uma vez que é a base com maior número de rótulos (53) e o *Subset Accuracy* exige fortemente que a predição do classificador acerte a única possível combinação dentre as 2^{53} possíveis. Mas o interessante é que, apesar do *Subset Accuracy* ser o pior para essa base, o *Hamming Loss* é o quarto melhor. Isso sugere que há alguns poucos rótulos que são difíceis de serem corretamente preditos. Para o classificador base KNN, o RDBR obteve o melhor resultado na métrica *Subset Accuracy* e na métrica *Example Based Accuracy*.

A ordem dos métodos que obtiverem o melhor rank médio altera-se consideravelmente ao alterar o classificador base. Por exemplo, com o KNN, a ordem crescente do rank médio no *Subset Accuracy* é RDBR,MCC,DBR,CC,BR,ECC, entretanto com o SVM a ordem mudou para ECC,MCC,RDBR,CC,DBR,BR.

O método BR alcança resultados melhores pela métrica *Hamming Loss* do que para as outras. O seu ranking médio é sempre menor quando medido pelo *Hamming Loss*. Isso sugere que, apesar do BR não capturar a dependência entre rótulos, ele captura bem a dependência que cada rótulo tem com o vetor de características. Uma hipótese para explicar o desempenho dos métodos multirrótulos que exploram correlação entre rótulos ser pior que o do BR em alguns casos, é que os métodos empenham muitos esforços para achar a dependência entre os rótulos e acabam desprezando a dependência de cada rótulo com o vetor de características. Por exemplo, se o problema multirrótulo tiver um número de rótulos relativamente muito maior que o número de características, a estratégia de expandir o espaço de características com os valores dos rótulos irá reduzir significativamente a importância do vetor de características. A hipótese é verdadeira quando não há (ou é baixa) a correlação entre rótulos na base de dados ou quando é complexa demais para ser entendida pelos métodos atuais.

Subset Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.5084(3)	0.5007(5)	0.5116(1)	0.4991(6)	0.5023(4)	0.5115(2)
Emotions	0.3101(5)	0.3406(1)	0.3119(4)	0.3(6)	0.3287(3)	0.3355(2)
Enron	0.0717(5)	0.0899(3)	0.1005(2)	0.077(4)	0.0664(6)	0.1022(1)
Genbase	0.9351(4)	0.9411(1)	0.9396(2)	0.0(6)	0.9336(5)	0.9381(3)
Medical	0.4519(6)	0.503(2)	0.5(3)	0.4796(5)	0.4847(4)	0.5133(1)
Motorpump	0.2857(1)	0.2799(4)	0.266(6)	0.2821(2)	0.2806(3)	0.2791(5)
Scene	0.6452(6)	0.668(3)	0.6498(5)	0.661(4)	0.6788(2)	0.7009(1)
Yeast	0.2209(6)	0.2454(3)	0.2226(5)	0.2301(4)	0.2615(1)	0.247(2)
Rank médio	4.5	2.75	3.5	4.625	3.5	2.125

Hamming Loss

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.0453(1)	0.0489(6)	0.0476(5)	0.0463(2)	0.0474(4)	0.0469(3)
Emotions	0.1937(1)	0.2041(3)	0.2159(6)	0.1973(2)	0.2063(4)	0.2068(5)
Enron	0.0581(1.5)	0.059(3.5)	0.0606(5)	0.0581(1.5)	0.059(3.5)	0.0607(6)
Genbase	0.0031(1)	0.0033(2)	0.0059(5)	1.0(6)	0.0038(4)	0.0034(3)
Medical	0.0175(3.5)	0.0165(1)	0.0188(5)	0.0171(2)	0.0175(3.5)	0.0189(6)
Motorpump	0.1618(2)	0.1697(3)	0.1864(6)	0.161(1)	0.1714(5)	0.171(4)
Scene	0.0925(2)	0.1003(5)	0.1065(6)	0.0931(3)	0.0965(4)	0.0906(1)
Yeast	0.1981(2)	0.2159(6)	0.2109(5)	0.195(1)	0.2054(4)	0.2031(3)
Rank médio	1.75	3.6875	5.375	2.3125	4	3.875

Example Based Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.58(1)	0.5689(2)	0.5656(3)	0.5555(6)	0.5648(5)	0.5651(4)
Emotions	0.5515(5)	0.5765(1)	0.5737(2)	0.545(6)	0.5604(4)	0.5691(3)
Enron	0.2344(4)	0.2456(3)	0.264(2)	0.2198(5)	0.2085(6)	0.265(1)
Genbase	0.9601(4)	0.9639(1)	0.9628(2)	0.0(6)	0.96(5)	0.9621(3)
Medical	0.5147(6)	0.5724(3)	0.5966(1)	0.5455(5)	0.5476(4)	0.5857(2)
Motorpump	0.5452(1)	0.5359(2)	0.5123(6)	0.5346(3)	0.5289(4)	0.5245(5)
Scene	0.6742(6)	0.6987(4)	0.7123(2)	0.692(5)	0.7089(3)	0.7322(1)
Yeast	0.524(4)	0.52(6)	0.5238(5)	0.5361(2)	0.5368(1)	0.5343(3)
Rank médio	3.875	2.75	2.875	4.75	4	2.75

Tabela 5.2: Desempenho dos métodos multirrótulos com *KNN* medidos pelas métricas *Subset Accuracy*,*Hamming Loss* e *Example Based Accuracy*

Subset Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.4775(2)	0.4496(6)	0.4528(4)	0.4604(3)	0.4822(1)	0.4527(5)
Emotions	0.2917(6)	0.3051(5)	0.3086(4)	0.3356(2)	0.3255(3)	0.3558(1)
Enron	0.0741(6)	0.1052(4)	0.094(5)	0.1334(1)	0.1087(2)	0.1075(3)
Genbase	0.9033(5)	0.9154(1)	0.9123(3)	0.8836(6)	0.9047(4)	0.9139(2)
Medical	0.5931(6)	0.6228(4.5)	0.6228(4.5)	0.6483(1)	0.6361(3)	0.6443(2)
Motorpump	0.2186(6)	0.2755(3)	0.2558(5)	0.293(1)	0.2748(4)	0.2763(2)
Scene	0.5322(6)	0.6394(3)	0.5422(5)	0.6572(1)	0.6548(2)	0.614(4)
Yeast	0.1489(6)	0.1965(3)	0.1535(5)	0.2006(2)	0.2193(1)	0.1845(4)
Rank médio	5.375	3.6875	4.4375	2.125	2.5	2.875

Hamming Loss

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.0596(2)	0.0651(4)	0.0662(6)	0.0567(1)	0.0624(3)	0.066(5)
Emotions	0.1917(2)	0.2134(5)	0.2156(6)	0.1847(1)	0.2108(4)	0.1929(3)
Enron	0.0799(6)	0.0744(3.5)	0.0744(3.5)	0.0529(1)	0.0726(2)	0.0762(5)
Genbase	0.0041(5)	0.0036(2)	0.0035(1)	0.0051(6)	0.004(4)	0.0037(3)
Medical	0.0127(6)	0.0126(4.5)	0.0125(2.5)	0.0112(1)	0.0125(2.5)	0.0126(4.5)
Motorpump	0.1657(2)	0.167(4)	0.1701(6)	0.154(1)	0.1685(5)	0.1659(3)
Scene	0.1066(3)	0.1067(4)	0.2094(6)	0.0903(1)	0.1029(2)	0.1167(5)
Yeast	0.2001(1)	0.2246(6)	0.221(5)	0.2027(2)	0.2108(4)	0.2103(3)
Rank médio	3.375	4.125	4.5	1.75	3.3125	3.9375

Example Based Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.6068(1)	0.586(4)	0.5786(6)	0.5975(3)	0.6041(2)	0.5828(5)
Emotions	0.5345(6)	0.5434(5)	0.5842(2)	0.5781(3)	0.5541(4)	0.6018(1)
Enron	0.3503(6)	0.3728(3)	0.3661(5)	0.4331(1)	0.382(2)	0.3688(4)
Genbase	0.9552(5)	0.9608(1)	0.9603(2)	0.9444(6)	0.9555(4)	0.9597(3)
Medical	0.6983(6)	0.713(4)	0.714(3)	0.7226(2)	0.7128(5)	0.727(1)
Motorpump	0.465(6)	0.5275(5)	0.5488(2)	0.5535(1)	0.5286(4)	0.5398(3)
Scene	0.6065(6)	0.6916(3)	0.6233(5)	0.7036(1)	0.7018(2)	0.6604(4)
Yeast	0.5064(4)	0.4921(6)	0.4949(5)	0.5243(2)	0.5258(1)	0.5142(3)
Rank médio	5	3.875	3.75	2.375	3	3

Tabela 5.3: Desempenho dos métodos multirrótulos com *SVM* medidos pelas métricas *Subset Accuracy*,*Hamming Loss* e *Example Based Accuracy*

Subset Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.4683(6)	0.4791(4)	0.49(2.5)	0.5318(1)	0.4775(5)	0.49(2.5)
Emotions	0.1637(6)	0.1888(4)	0.1822(5)	0.3002(1)	0.2328(2)	0.2244(3)
Enron	0.1034(5)	0.1257(2)	0.0975(6)	0.1381(1)	0.114(3)	0.1116(4)
Genbase	0.9714(2.5)	0.9699(5)	0.9714(2.5)	0.9623(6)	0.9714(2.5)	0.9714(2.5)
Medical	0.6718(6)	0.6902(4)	0.6974(2)	0.6739(5)	0.6932(3)	0.7035(1)
Motorpump	0.2274(6)	0.2536(3)	0.2376(5)	0.3185(1)	0.2442(4)	0.2573(2)
Scene	0.4408(5)	0.5692(2)	0.4366(6)	0.5962(1)	0.5501(3)	0.543(4)
Yeast	0.0658(6)	0.1448(2)	0.0666(5)	0.1684(1)	0.1303(3)	0.1212(4)
Rank médio	5.3125	3.25	4.25	2.125	3.1875	2.875

Hamming Loss

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.0517(5)	0.0501(2)	0.051(3.5)	0.0415(1)	0.052(6)	0.051(3.5)
Emotions	0.2529(2)	0.2693(5)	0.2729(6)	0.1945(1)	0.2575(3)	0.2645(4)
Enron	0.0509(2)	0.054(5)	0.0559(6)	0.049(1)	0.0532(3)	0.0537(4)
Genbase	0.0012(2.5)	0.0013(5)	0.0012(2.5)	0.0016(6)	0.0012(2.5)	0.0012(2.5)
Medical	0.01(5)	0.0097(3.5)	0.0093(1)	0.0101(6)	0.0097(3.5)	0.0095(2)
Motorpump	0.1753(3)	0.174(2)	0.1791(6)	0.1398(1)	0.1769(4)	0.1782(5)
Scene	0.1307(2)	0.1334(3)	0.1779(6)	0.0941(1)	0.1378(4)	0.1449(5)
Yeast	0.2489(2)	0.268(4)	0.2829(6)	0.2046(1)	0.2702(5)	0.2637(3)
Rank médio	2.9375	3.6875	4.625	2.25	3.875	3.625

Example Based Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.5628(6)	0.5706(4)	0.5764(3)	0.5935(1)	0.5629(5)	0.5777(2)
Emotions	0.4447(4)	0.4405(5)	0.4176(6)	0.5238(1)	0.469(2)	0.4631(3)
Enron	0.4131(4)	0.4115(5)	0.4141(3)	0.44(1)	0.3989(6)	0.4163(2)
Genbase	0.9854(2.5)	0.9847(5)	0.9854(2.5)	0.9779(6)	0.9854(2.5)	0.9854(2.5)
Medical	0.7585(5)	0.7707(4)	0.7822(2)	0.7573(6)	0.7732(3)	0.7845(1)
Motorpump	0.5252(6)	0.549(2)	0.5347(5)	0.6046(1)	0.5393(4)	0.5444(3)
Scene	0.54(5)	0.6205(2)	0.5336(6)	0.6276(1)	0.6011(3)	0.5843(4)
Yeast	0.4357(3)	0.4252(4)	0.4171(6)	0.4932(1)	0.4196(5)	0.4451(2)
Rank médio	4.4375	3.875	4.1875	2.25	3.8125	2.4375

Tabela 5.4: Desempenho dos métodos multirrótulos com *C4.5* medidos pelas métricas *Subset Accuracy*,*Hamming Loss* e *Example Based Accuracy*

Subset Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.4403(6)	0.4512(2)	0.4434(5)	0.4836(1)	0.448(3.5)	0.448(3.5)
Emotions	0.2343(5)	0.268(1)	0.2073(6)	0.2561(2)	0.2411(4)	0.2427(3)
Enron	0.1093(3)	0.1134(2)	0.1017(6)	0.1257(1)	0.1081(4)	0.107(5)
Genbase	0.9562(4)	0.9532(5)	0.9577(2)	0.7901(6)	0.9577(2)	0.9577(2)
Medical	0.4499(6)	0.4653(5)	0.4684(4)	0.5297(1)	0.4745(3)	0.4827(2)
Motorpump	0.2857(6)	0.3003(3)	0.2944(5)	0.3039(1)	0.2995(4)	0.301(2)
Scene	0.4989(5)	0.6003(2)	0.4944(6)	0.5941(3)	0.6061(1)	0.5845(4)
Yeast	0.1369(6)	0.1841(2)	0.1386(5)	0.1709(3)	0.1907(1)	0.168(4)
Rank médio	5.125	2.75	4.875	2.25	2.8125	3.1875

Hamming Loss

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.0685(5)	0.0688(6)	0.067(2)	0.0554(1)	0.0675(4)	0.0673(3)
Emotions	0.2134(2)	0.2308(3)	0.2485(6)	0.212(1)	0.2313(4)	0.2392(5)
Enron	0.0619(2)	0.0623(4)	0.0636(5)	0.0525(1)	0.0621(3)	0.0637(6)
Genbase	0.0023(3)	0.0025(5)	0.0022(2)	0.0084(6)	0.0021(1)	0.0024(4)
Medical	0.0217(5.5)	0.0217(5.5)	0.0212(3.5)	0.0155(1)	0.0212(3.5)	0.021(2)
Motorpump	0.1548(3)	0.1557(4.5)	0.1558(6)	0.1502(1)	0.154(2)	0.1557(4.5)
Scene	0.1086(2)	0.1145(4)	0.1603(6)	0.0986(1)	0.1109(3)	0.1287(5)
Yeast	0.2081(1)	0.2289(6)	0.2281(5)	0.2118(2)	0.2236(4)	0.2208(3)
Rank médio	2.9375	4.75	4.4375	1.75	3.0625	4.0625

Example Based Accuracy

Dataset	BR	CC	DBR	ECC	MCC	RDBR
Birds	0.5542(6)	0.5591(5)	0.5636(3)	0.5874(1)	0.5601(4)	0.5646(2)
Emotions	0.5011(4)	0.5103(1)	0.4984(6)	0.5092(2)	0.5041(3)	0.5006(5)
Enron	0.38(5)	0.3831(3)	0.3848(2)	0.4136(1)	0.3825(4)	0.3789(6)
Genbase	0.976(5)	0.9764(4)	0.979(1)	0.8967(6)	0.9786(3)	0.9787(2)
Medical	0.5903(6)	0.5951(5)	0.6061(3)	0.6504(1)	0.6055(4)	0.6143(2)
Motorpump	0.558(6)	0.5649(5)	0.5731(2)	0.5723(3)	0.5717(4)	0.5746(1)
Scene	0.5665(6)	0.6457(2)	0.5987(5)	0.6391(3)	0.6536(1)	0.6216(4)
Yeast	0.4966(2)	0.4762(6)	0.4782(5)	0.4989(1)	0.4945(3)	0.4833(4)
Rank médio	5	3.875	3.375	2.25	3.25	3.25

Tabela 5.5: Desempenho dos métodos multirrótulos com *Regressão Logística* medidos pelas métricas *Subset Accuracy*,*Hamming Loss* e *Example Based Accuracy*

5.2.2 Estudo Geral

As tabelas 5.6,5.7 e 5.8 mostram o resultado alcançado por cada combinação de modelo de classificação multirrótulo e classificador base nas métricas *Subset Accuracy*,*Hamming Loss* e *Example Based Accuracy*. O ECC com KNN foi o único que não alcançou nenhum resultado, pois precisou de mais memória principal do que tinha disponível e é por isso que temos nas tabelas o símbolo do ponto de interrogação em seu resultado. A esse atribuímos a última colocação na base correpondente. Observando as tabelas podemos notar que os método multirrótulo ECC obtém os melhores resultados em geral. O ECC com C4.5 obteve o melhor rank médio para as métricas *Subset Accuracy* e *Hamming Loss*, e o segundo melhor rank médio para a métrica *Example Based Accuracy*. O RDBR com KNN alcançou resultados bons, ele teve o segundo melhor rank médio pela métrica *Subset Accuracy*. Interessante que o RDBR com KNN obtém resultados muito melhores do que com outros classificadores base. Isso não ocorre com outros métodos, visto que alguns alcançam resultados melhores usando SVM e outros com C4.5. Note que o *rank* de cada um dos métodos variam muito de base de dados para base de dados, ou seja, desvio padrão dos *rank* é alto. Por exemplo, o MCC com KNN alcançou o pior desempenho na base *Enron*, porém teve o melhor desempenho na *Yeast*. Esse desvio padrão alto sugere que não há método multirrótulo com um classificador base único que seja bom para todos os problemas, cada um é bom para um tipo de problema.

Subset Accuracy

	Birds	**Emotions**	**Enron**	**Genbase**	**Medical**	**Motor**	**Scene**	**Yeast**	**Rank Médio**	**Desvio Padrão**
ECC (C4.5)	.531(1)	.300(11.5)	.138(1)	.962(6)	.673(5)	.318(1)	.596(13)	.168(14.5)	6.63	5.66
RDBR (KNN)	.511(2.5)	.335(3.5)	.102(15)	.938(14)	.513(14)	.279(12.5)	.700(1)	.247(2)	8.06	6.29
ECC (SVM)	.460(16)	.335(3.5)	.133(2)	.883(22)	.648(7)	.293(7)	.657(5)	.200(8)	8.81	6.78
CC (KNN)	.500(6)	.340(2)	.089(20)	.941(12)	.503(15)	.279(12.5)	.668(3)	.245(3)	9.19	6.63
MCC (KNN)	.502(5)	.328(5)	.066(24)	.933(16)	.484(17)	.280(11)	.678(2)	.261(1)	10.13	8.25
DBR (KNN)	.511(2.5)	.311(7)	.100(17)	.939(13)	.500(16)	.266(17)	.649(7)	.222(5)	10.56	5.85
MCC (SVM)	.482(11)	.325(6)	.108(9.5)	.904(20)	.636(9)	.274(16)	.654(6)	.219(7)	10.56	5.02
RDBR (SVM)	.452(17.5)	.355(1)	.107(11.5)	.913(18)	.644(8)	.276(14)	.614(10)	.184(11.5)	11.44	5.47
ECC (Logi)	.483(10)	.256(15)	.125(3.5)	.790(23)	.529(13)	.303(2)	.594(14)	.170(13)	11.69	6.67
BR (KNN)	.508(4)	.310(8)	.071(23)	.935(15)	.451(23)	.285(8.5)	.645(8)	.220(6)	11.94	7.51
RDBR (C4.5)	.490(8.5)	.224(20)	.111(7)	.971(2.5)	.703(1)	.257(18)	.543(18)	.121(22)	12.13	8.32
CC (Logi)	.451(19)	.268(14)	.113(6)	.953(11)	.465(22)	.300(4)	.600(12)	.184(11.5)	12.44	6.01
CC (C4.5)	.479(12)	.188(22)	.125(3.5)	.969(5)	.690(4)	.253(20)	.569(16)	.144(18)	12.56	7.55
ECC (KNN)	.499(7)	.300(11.5)	.077(21)	.???(24)	.479(19)	.282(10)	.661(4)	.230(4)	12.56	7.83
MCC (C4.5)	.477(13.5)	.232(19)	.114(5)	.971(2.5)	.693(3)	.244(21)	.550(17)	.130(21)	12.75	8.05
MCC (Logi)	.448(21.5)	.241(17)	.108(9.5)	.957(8)	.474(20)	.299(5)	.606(11)	.190(10)	12.75	5.99
CC (SVM)	.449(20)	.305(10)	.105(13)	.915(17)	.622(10.5)	.275(15)	.639(9)	.196(9)	12.94	4.07
RDBR (Logi)	.448(21.5)	.242(16)	.107(11.5)	.957(8)	.482(18)	.301(3)	.584(15)	.168(14.5)	13.44	5.83
DBR (C4.5)	.490(8.5)	.182(23)	.097(18)	.971(2.5)	.697(2)	.237(22)	.436(24)	.066(23)	15.38	9.51
DBR (SVM)	.452(17.5)	.308(9)	.094(19)	.912(19)	.622(10.5)	.255(19)	.542(19)	.153(16)	16.13	4.09
BR (C4.5)	.468(15)	.163(24)	.103(14)	.971(2.5)	.671(6)	.227(23)	.440(23)	.065(24)	16.44	8.55
BR (Logi)	.440(24)	.234(18)	.109(8)	.956(10)	.449(24)	.285(8.5)	.498(21)	.136(20)	16.69	6.82
DBR (Logi)	.443(23)	.207(21)	.101(16)	.957(8)	.468(21)	.294(6)	.494(22)	.138(19)	17.00	6.55
BR (SVM)	.477(13.5)	.291(13)	.074(22)	.903(21)	.593(12)	.218(24)	.532(20)	.148(17)	17.81	4.58

Tabela 5.6: Desempenho em *Subset Accuracy* de cada combinação de método multirrótulo e classificador base. Regressão Logística é abreviado por *Logi*.

Hamming Loss

	Birds	**Emotions**	**Enron**	**Genbase**	**Medical**	**Motor**	**Scene**	**Yeast**	**Rank Médio**	**Desvio Padrão**
ECC (C4.5)	.041(1)	.194(5)	.049(1)	.001(3.5)	.010(5.5)	.139(1)	.094(5)	.204(6)	3.50	2.19
ECC (SVM)	.056(14)	.184(1)	.052(3.5)	.005(21.5)	.011(7)	.154(4)	.090(1.5)	.202(4)	7.06	7.13
BR (KNN)	.045(2)	.193(4)	.058(9.5)	.003(15)	.017(16)	.161(9.5)	.092(3)	.198(2)	7.63	5.72
ECC (KNN)	.046(3.5)	.197(6)	.058(9.5)	.???(24)	.017(16)	.161(9.5)	.093(4)	.195(1)	9.19	7.58
RDBR (KNN)	.046(3.5)	.206(8.5)	.060(13.5)	.003(15)	.018(18.5)	.171(17.5)	.090(1.5)	.203(5)	10.38	6.62
ECC (Logi)	.055(13)	.212(11)	.052(3.5)	.008(23)	.015(13)	.150(2)	.098(7)	.211(12)	10.56	6.6
MCC (KNN)	.047(5.5)	.206(8.5)	.059(11.5)	.003(15)	.017(16)	.171(17.5)	.096(6)	.205(7)	10.88	4.79
CC (KNN)	.048(7)	.204(7)	.059(11.5)	.003(15)	.016(14)	.169(15)	.100(8)	.215(13)	11.31	3.49
BR (SVM)	.059(15)	.191(2)	.079(24)	.004(19.5)	.012(10)	.165(11.5)	.106(11)	.200(3)	12.00	7.52
BR (C4.5)	.051(10)	.252(20)	.050(2)	.001(3.5)	.010(5.5)	.175(20)	.130(18)	.248(20)	12.38	7.98
CC (C4.5)	.050(8)	.269(23)	.054(7)	.001(3.5)	.009(2.5)	.174(19)	.133(19)	.268(22)	13.00	8.57
BR (Logi)	.068(23.5)	.213(12.5)	.061(15)	.002(9)	.021(22)	.154(4)	.108(13)	.208(8)	13.38	6.72
RDBR (SVM)	.066(18.5)	.192(3)	.076(23)	.003(15)	.012(10)	.165(11.5)	.116(16)	.210(10)	13.38	6.12
RDBR (C4.5)	.051(10)	.264(22)	.053(5.5)	.001(3.5)	.009(2.5)	.178(22)	.144(21)	.263(21)	13.44	8.9
MCC (C4.5)	.052(12)	.257(21)	.053(5.5)	.001(3.5)	.009(2.5)	.176(21)	.137(20)	.270(23)	13.56	8.72
MCC (SVM)	.062(16)	.210(10)	.072(20)	.004(19.5)	.012(10)	.168(14)	.102(9)	.210(10)	13.56	4.5
CC (SVM)	.065(17)	.213(12.5)	.074(21.5)	.003(15)	.012(10)	.167(13)	.106(11)	.224(17)	14.63	3.79
DBR (C4.5)	.051(10)	.272(24)	.055(8)	.001(3.5)	.009(2.5)	.179(23)	.177(23)	.282(24)	14.75	9.65
DBR (KNN)	.047(5.5)	.215(14.5)	.060(13.5)	.005(21.5)	.018(18.5)	.186(24)	.106(11)	.210(10)	14.81	6.2
MCC (Logi)	.067(21)	.231(17)	.062(16.5)	.002(9)	.021(22)	.154(4)	.110(14)	.223(16)	14.94	5.98
RDBR (Logi)	.067(21)	.239(18)	.063(18.5)	.002(9)	.021(22)	.155(7)	.128(17)	.220(14)	15.81	5.42
CC (Logi)	.068(23.5)	.230(16)	.062(16.5)	.002(9)	.021(22)	.155(7)	.114(15)	.228(18.5)	15.94	5.73
DBR (SVM)	.066(18.5)	.215(14.5)	.074(21.5)	.003(15)	.012(10)	.170(16)	.209(24)	.221(15)	16.81	4.4
DBR (Logi)	.067(21)	.248(19)	.063(18.5)	.002(9)	.021(22)	.155(7)	.160(22)	.228(18.5)	17.13	5.84

Tabela 5.7: Desempenho em *Hamming Loss* de cada combinação de método multirrótulo e classificador base. Regressão Logística foi abreviado por *Logi*.

Example Based Accuracy

	Birds	**Emotions**	**Enron**	**Genbase**	**Medical**	**Motor**	**Scene**	**Yeast**	**Rank Médio**	**Desvio Padrão**
ECC (SVM)	.597(3)	.578(3)	.433(2)	.944(22)	.722(8)	.553(8)	.703(4)	.524(5.5)	6.94	6.49
ECC (C4.5)	.593(4)	.523(13)	.440(1)	.977(9)	.757(6)	.604(1)	.627(14)	.493(15)	7.88	5.72
MCC (SVM)	.604(2)	.554(8)	.382(11.5)	.955(20.5)	.712(11)	.528(18.5)	.701(5)	.525(4)	10.06	6.71
RDBR (SVM)	.582(7)	.601(1)	.368(16)	.959(19)	.727(7)	.539(13.5)	.660(10)	.514(9)	10.31	5.71
ECC (Logi)	.587(5)	.509(15)	.413(5.5)	.896(23)	.650(13)	.572(4)	.639(13)	.498(11)	11.19	6.36
RDBR (C4.5)	.577(10)	.463(21)	.416(3)	.985(2.5)	.784(1)	.544(12)	.584(21)	.445(20)	11.31	8.61
DBR (SVM)	.578(9)	.584(2)	.366(17)	.960(16.5)	.714(9)	.548(10)	.623(15)	.494(13.5)	11.50	5.02
CC (C4.5)	.570(12)	.440(23)	.411(7)	.984(5)	.770(4)	.549(9)	.620(17)	.425(22)	12.38	7.48
RDBR (KNN)	.565(14.5)	.569(6)	.265(19)	.962(13.5)	.585(20)	.524(22)	.732(1)	.534(3)	12.38	8.09
CC (KNN)	.568(13)	.576(4)	.245(21)	.963(12)	.572(21)	.535(15)	.698(6)	.520(8)	12.50	6.39
MCC (Logi)	.560(21)	.504(16)	.382(11.5)	.978(7.5)	.605(16)	.571(5)	.653(11)	.494(13.5)	12.69	5.1
DBR (KNN)	.565(14.5)	.573(5)	.264(20)	.962(13.5)	.596(17)	.512(23)	.712(2)	.523(7)	12.75	7.45
CC (SVM)	.586(6)	.543(11)	.372(15)	.960(16.5)	.713(10)	.527(20)	.691(8)	.492(16)	12.81	4.8
BR (KNN)	.580(8)	.551(9)	.234(22)	.960(16.5)	.514(24)	.545(11)	.674(9)	.524(5.5)	13.13	6.88
RDBR (Logi)	.564(16.5)	.500(18)	.378(14)	.978(7.5)	.614(14)	.574(2)	.621(16)	.483(17)	13.13	5.55
DBR (C4.5)	.576(11)	.417(24)	.414(4)	.985(2.5)	.782(2)	.534(16.5)	.533(24)	.417(24)	13.50	9.92
DBR (Logi)	.563(18)	.498(19)	.384(9)	.979(6)	.606(15)	.573(3)	.598(20)	.478(18)	13.50	6.57
MCC (C4.5)	.562(19.5)	.469(20)	.398(8)	.985(2.5)	.773(3)	.539(13.5)	.601(19)	.419(23)	13.56	8.11
MCC (KNN)	.564(16.5)	.560(7)	.208(24)	.960(16.5)	.547(22)	.528(18.5)	.708(3)	.536(1.5)	13.63	8.63
CC (Logi)	.559(22)	.510(14)	.383(10)	.976(10.5)	.595(18)	.564(6)	.645(12)	.476(19)	13.94	5.36
BR (SVM)	.606(1)	.534(12)	.350(18)	.955(20.5)	.698(12)	.465(24)	.606(18)	.506(10)	14.44	7.23
BR (C4.5)	.562(19.5)	.444(22)	.413(5.5)	.985(2.5)	.758(5)	.525(21)	.540(23)	.435(21)	14.94	8.88
BR (Logi)	.554(24)	.501(17)	.380(13)	.976(10.5)	.590(19)	.558(7)	.566(22)	.496(12)	15.56	5.91
ECC (KNN)	.555(23)	.545(10)	.219(23)	.???(24)	.545(23)	.534(16.5)	.692(7)	.536(1.5)	16.00	8.77

Tabela 5.8: Desempenho em *Example Based Accuracy* de cada combinação de método multirrótulo e classificador base

6 Conclusão

Este trabalho teve por objetivo as seguintes atividades:

1. Descobrir como medir e explorar correlação entre rótulos;
2. Análise crítica dos métodos multirrótulos;
3. Elaboração de um algoritmo de um novo método multirrótulo.

O objetivo 1 foi considerado alcançado ao verificarmos que a métrica *Subset Accuracy* mede a exploração da correlação entre rótulos e também ao verificarmos que a estratégia adotada por vários métodos que se baseia na expansão do espaço de características com valores de rótulos, realmente faz com que o modelo de classificação considera dependência entre rótulos, mesmo que parcialmente. O objetivo 2 foi considerado alcançado pelas análises feita na seção 5.2. O objetivo 3 foi considerado alcançado, pois desenvolveu-se um novo modelo de classificação, chamado de Recursive Dependent Binary Relevance, que apresenta resultados melhores que alguns métodos multirrótulo da literatura. Vale ressaltar que o novo método apresentou sempre resultados experimentais melhores que o método na qual ele foi baseado, o DBR. Isso comprova que o novo método é uma melhoria do antigo.

Adicionalmente, a partir dos experimentos mostrados e analisados na seção 5.2 podemos chegar aos seguintes pontos conclusivos:

- o melhor classificador base para um método multirrótulo, aquele que resulta no maior desempenho em uma métrica, não é necessariamente o melhor para os outros métodos multirrótulos.
- Para problemas multirrótulo cuja dependência entre rótulos é baixa, inexistente ou complexa, o método *Binary Relevance* apresenta resultados melhores que os métodos multirrótulo estudados.

- A métrica *Subset Accuracy* tende a ser alto para métodos que exploram correlação entre rótulos, e o contrário para o *Hamming Loss*.

Bibliografia

BORGES, H. B. *Classificador Hierárquico Multirrótulo Usando Uma Rede Neural Competitiva*. Tese (Doutorado em Informática) — Universidade Católica do Paraná, 2012.

CARVALHO, A.; FREITAS, A. A tutorial on multi-label classification techniques. In: ABRAHAM, A.; HASSANIEN, A.-E.; SNÁSEL, V. (Ed.). *Foundations of Computational Intelligence Volume 5*. : Springer Berlin Heidelberg, 2009, (Studies in Computational Intelligence, v. 205). p. 177–195. ISBN 978-3-642-01535-9.

DEMBCZYNSKI, K.; CHENG, W.; HÜLLERMEIER, E. Bayes optimal multilabel classification via probabilistic classifier chains. In: FÜRNKRANZ, J.; JOACHIMS, T. (Ed.). *ICML*. : Omnipress, 2010. p. 279–286. ISBN 978-1-60558-907-7.

DICKMAN, B.; GILMAN, M. Monte carlo optimization. *Journal of Optimization Theory and Applications*, Kluwer Academic Publishers-Plenum Publishers, v. 60, n. 1, p. 149–157, 1989. ISSN 0022-3239.

DUNHAM, M. H.; MING, D. *Introductory and Advanced Topics*. : Prentice Hall, 2003.

HALL, M. et al. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, ACM, v. 11, n. 1, p. 10–18, 2009.

JAMES, G. et al. *An introduction to statistical learning*. : Springer, 2013.

MENDEL, E. et al. Automatic bearing fault pattern recognition using vibration signal analysis. In: *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*. [S.l.: s.n.], 2008. p. 955–960.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. : MIT Press, 2012.

MONTANES, E. et al. Dependent binary relevance models for multi-label classification. *Pattern Recognition*, v. 47, n. 3, p. 1494 – 1508, 2014. ISSN 0031-3203. Handwriting Recognition and other {PR} Applications. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0031320313004019>.

PENG, W.; CHEN, J.; ZHOU, H. An implementation of id3—decision tree learning algorithm. 2009.

READ, J.; MARTINO, L.; LUENGO, D. Efficient monte carlo optimization for multi-label classifier chains. *CoRR*, abs/1211.2190, 2012.

READ, J. et al. Classifier chains for multi-label classification. In: *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*. Berlin, Heidelberg: Springer-Verlag, 2009. (ECML PKDD '09), p. 254–269. ISBN 978-3-642-04173-0.

REZENDE, S. *Sistemas inteligentes: fundamentos e aplicações*. : Manole, 2003. ISBN 9788520416839.

TSOUMAKAS, G.; KATAKIS, I.; VLAHAVAS, I. Mining multi-label data. In: MAIMON, O.; ROKACH, L. (Ed.). *Data Mining and Knowledge Discovery Handbook*. : Springer US, 2010. p. 667–685. ISBN 978-0-387-09822-7.

WU, X. et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, Springer, v. 14, n. 1, p. 1–37, 2008.

YU, C. H. Resampling methods: concepts, applications, and justification. *Practical Assessment, Research & Evaluation*, McGraw-Hill, v. 8, n. 19, p. 1–23, 2003.

ZHANG, M.-L.; ZHOU, Z.-H. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, IEEE Computer Society, Los Alamitos, CA, USA, v. 99, n. PrePrints, p. 1, 2013. ISSN 1041-4347.