Redes neurais e algoritmos genéticos para
problemas de classificação hierárquica
multirrótulo

Ricardo Cerri

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura:

Redes neurais e algoritmos genéticos para problemas de classificação hierárquica multirrótulo

Ricardo Cerri

***Orientador:* Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho**

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA.*

USP – São Carlos
Janeiro de 2014

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

C417r Cerri, Ricardo
 Redes neurais e algoritmos genéticos para
 problemas de classificação hierárquica multirrótulo /
 Ricardo Cerri; orientador André C. P. L. F. de
 Carvalho. -- São Carlos, 2014.
 175 p.

 Tese (Doutorado - Programa de Pós-Graduação em
 Ciências de Computação e Matemática Computacional) --
 Instituto de Ciências Matemáticas e de Computação,
 Universidade de São Paulo, 2014.

 1. Classificação Hierárquica Multirrótulo. 2.
 Aprendizado de Máquina. 3. Bioinformática. 4. Redes
 Neurais. 5. Algoritmos Genéticos. I. de Carvalho,
 André C. P. L. F., orient. II. Título.

Esta tese foi preparada com o formatador de textos $\text{\LaTeX}$. A bibliografia é gerada automaticamente pelo $\text{\BibTeX}$, utilizando o estilo *apalike* com modificações para o português. O estilo utilizado no documento e as modificações no estilo *apalike* foram desenvolvidas por Ronaldo Cristiano Prati.

A meus pais.

Agradecimentos

Agradeço a Deus por mais esta conquista em minha vida.

Agradeço muito aos meus pais Irineu e Cleide, que são e sempre foram meus grandes amigos e incentivadores. Sem eles eu não estaria aqui hoje. Sem eles eu não teria as oportunidades que tive na vida, e não teria feito esta Tese de Doutorado. Agradeço também ao meu irmão Rodrigo pela amizade e incentivo, e também a toda a minha família.

Ao meu orientador Prof. Dr. André Carlos P. L. F. de Carvalho pela excelente orientação durante todo o período de meu Doutorado. Seus conhecimentos e aconselhamentos foram fundamentais para a realização deste trabalho.

Quero também agradecer aos professores Alex A. Freitas e Yaochu Jin pela orientação durante os períodos que passei no exterior. Suas sugestões contribuíram muito para o desenvolvimento desta pesquisa.

Agradeço à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelos auxílios financeiros indispensáveis à realização deste trabalho (Processo n. 2009/17401-2 e Processo n. 2011/22321-8), e também à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo auxílio financeiro concedido no início desta pesquisa.

Ao meu amigo Rodrigo Barros pela colaboração e realização de trabalhos em conjunto, e também aos amigos Danilo Horta e Thiago Covões pela troca de ideias e ajuda em vários momentos do Doutorado. Também agradeço a todos os amigos do Laboratório de Computação Bioinspirada do ICMC/USP, Jonathan Andrade, Débora Medeiros, Pablo Jaskowiak, Luiz Coletta, André Rossi, Paulo Gabriel, Rosane Maffei, Lucas Vendramin, e também aos amigos Roberto Yokoyama, Renato Rodrigues, Thiago Siqueira e vários outros amigos. Agradeço a todos pela troca de ideias e pelos momentos agradáveis que passamos juntos.

A Leander Schietgat e Celine Vens pela ajuda com os métodos baseados em *Predictive Clustering Trees*, e também a Fernando Otero pela ajuda com o método *hmAnt-Miner*. Também a Mauri Ferrandin e Helyane Borges pela troca de ideias em vários momentos da pesquisa.

Agradeço também a todos os professores e funcionários do ICMC/USP - São Carlos, que contribuíram diretamente ou indiretamente para a realização deste trabalho.

Resumo

Em problemas convencionais de classificação, cada exemplo de um conjunto de dados é associado a apenas uma dentre duas ou mais classes. No entanto, existem problemas de classificação mais complexos, nos quais as classes envolvidas no problema são estruturadas hierarquicamente, possuindo subclasses e superclasses. Nesses problemas, exemplos podem ser atribuídos simultaneamente a classes pertencentes a dois ou mais caminhos de uma hierarquia, ou seja, exemplos podem ser classificados em várias classes localizadas em um mesmo nível hierárquico. Tal hierarquia pode ser estruturada como uma árvore ou como um grafo acíclico direcionado. Esses problemas são chamados de problemas de classificação hierárquica multirrótulo, sendo mais difíceis devido à alta complexidade, diversidade de soluções, difícil modelagem e desbalanceamento dos dados. Duas abordagens são utilizadas para tratar esses problemas, chamadas global e local. Na abordagem global, um único classificador é induzido para lidar com todas as classes do problema simultaneamente, e a classificação de novos exemplos é realizada em apenas um passo. Já na abordagem local, um conjunto de classificadores é induzido, sendo cada classificador responsável pela predição de uma classe ou de um conjunto de classes, e a classificação de novos exemplos é realizada em vários passos, considerando as predições dos vários classificadores. Nesta Tese de Doutorado, são propostos e investigados dois métodos para classificação hierárquica multirrótulo. O primeiro deles é baseado na abordagem local, e associa uma rede neural *Multi-Layer Perceptron* (MLP) a cada nível da hierarquia, sendo cada MLP responsável pelas predições no seu nível associado. O método é chamado *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* (HMC-LMLP). O segundo método é baseado na abordagem global, e induz regras de classificação hierárquicas multirrótulo utilizando um Algoritmo Genético. O método é chamado *Hierarchical Multi-Label Classification with a Genetic Algorithm* (HMC-GA). Experimentos utilizando hierarquias estruturadas como árvores mostraram que o método HMC-LMLP obteve desempenhos de classificação superiores ao método estado-da-arte na literatura, e desempenhos superiores ou competitivos quando utilizando hierarquias estruturadas como grafos. O método HMC-GA obteve resultados competitivos com outros métodos da literatura em hierarquias estruturadas como árvores e grafos, sendo capaz de induzir, em muitos casos, regras menores e em menor quantidade.

Abstract

In conventional classification problems, each example of a dataset is associated with just one among two or more classes. However, there are more complex classification problems where the classes are hierarchically structured, having subclasses and superclasses. In these problems, examples can be simultaneously assigned to classes belonging to two or more paths of a hierarchy, *i.e.*, examples can be classified in many classes located in the same hierarchical level. Such a hierarchy can be structured as a tree or a directed acyclic graph. These problems are known as hierarchical multi-label classification problems, being more difficult due to the high complexity, diversity of solutions, modeling difficulty and data imbalance. Two main approaches are used to deal with these problems, called global and local. In the global approach, only one classifier is induced to deal with all classes simultaneously, and the classification of new examples is done in just one step. In the local approach, a set of classifiers is induced, where each classifier is responsible for the predictions of one class or a set of classes, and the classification of new examples is done in many steps, considering the predictions of all classifiers. In this Thesis, two methods for hierarchical multi-label classification are proposed and investigated. The first one is based on the local approach, and associates a *Multi-Layer Perceptron* (MLP) to each hierarchical level, being each MLP responsible for the predictions in its associated level. The method is called *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* (HMC-LMLP). The second method is based on the global approach, and induces hierarchical multi-label classification rules using a Genetic Algorithm. The method is called *Hierarchical Multi-Label Classification with a Genetic Algorithm* (HMC-GA). Experiments using hierarchies structured as trees showed that HMC-LMLP obtained classification performances superior to the state-of-the-art method in the literature, and superior or competitive performances when using graph-structured hierarchies. The HMC-GA method obtained competitive results with other methods of the literature in both tree and graph-structured hierarchies, being able of inducing, in many cases, smaller and in less quantity rules.

Sumário

Lista de Abreviaturas	xix
Lista de Figuras	xxi
Lista de Tabelas	xxiii
Lista de Algoritmos	xxvii
1 Introdução	1
1.1 Motivação	2
1.2 Métodos Propostos	4
1.3 Hipótese e Objetivo	5
1.4 Resumo dos Resultados	7
1.5 Organização do Documento	7
2 Conceitos Fundamentais de Redes Neurais e Computação Evolutiva	9
2.1 Computação Evolutiva	9
2.1.1 Algoritmo Evolutivo Tradicional	10
2.1.2 Algoritmos Genéticos	11
2.2 Redes Neurais Artificiais	15
2.2.1 O Sistema Nervoso	16
2.2.2 Neurônios e Sinapses	16
2.2.3 Neurônios Artificiais	17
2.2.4 Single-Layer Perceptron	20
2.2.5 Multi-Layer Perceptron e Back-propagation	21
2.2.6 Resilient Back-propagation	26
2.3 Considerações Finais	28
3 Classificação Hierárquica e Multirrótulo	29
3.1 Classificação Hierárquica Simples-Rótulo	29
3.2 Classificação Multirrótulo não Hierárquica	32

3.3	Classificação Hierárquica Multirrótulo	34
3.4	Trabalhos Relacionados	35
3.4.1	Abordagens que não são Bioinspiradas	36
3.4.2	Abordagens Bioinspiradas	45
3.5	Revisão de Medidas de Avaliação	48
3.5.1	Medidas de Avaliação Baseadas em Hierarquia	49
3.5.2	Medidas de Avaliação Baseadas em Distância	54
3.5.3	Medidas de Avaliação Baseadas em Semântica	57
3.5.4	Contrastando Diferentes Medidas de Avaliação	59
3.6	Considerações Finais	62
4	Classificação Hierárquica Multirrótulo com Perceptrons Multi-Camadas Locais	63
4.1	Predições como Única Entrada	65
4.2	Predições como Complementação dos Vetores de Atributos	66
4.3	Classes Verdadeiras como Complementação dos Vetores de Atributos	69
4.4	Vetores de Atributos como Única Entrada	70
4.5	Obtendo as Predições Finais	70
4.6	Complexidade Computacional	72
4.7	Considerações Finais	72
5	Classificação Hierárquica Multirrótulo com um Algoritmo Genético	75
5.1	Representação dos Indivíduos	77
5.2	Inicialização da População	78
5.3	Evolução	80
5.4	Construção dos Consequentes e Cálculo do Fitness	88
5.5	Complexidade Computacional	89
5.6	Considerações Finais	91
6	Experimentos	93
6.1	Medidas de Avaliação	93
6.2	Métodos da Literatura Utilizados para Comparação	95
6.3	Conjuntos de Dados	96
6.4	Experimentos com o Método HMC-LMLP	100
6.4.1	Parâmetros	101
6.4.2	Experimentos	105
6.5	Experimentos com o Método HMC-GA	114
6.5.1	Parâmetros	115
6.5.2	Experimentos	116
6.6	Gráficos das curvas precisão-revocação	123
6.7	Considerações Finais	127

7	Conclusões	129
7.1	Contribuições da Pesquisa	131
7.2	Principais Resultados	133
7.3	Trabalhos Futuros	135
	Referências Bibliográficas	137
A	Resultados por Nível das Variações do método HMC-LMLP	151
B	Resultados por Nível das Variações Bp-Predicted, Rp-Predicted e Métodos da Literatura	159
C	Resultados das Variações Bp-Predicted, Rp-Predicted e Métodos da Literatura em Classes Específicas	165

Lista de Abreviaturas

ACO *Ant Colony Optimization*

AD *Árvores de Decisão*

AG *Algoritmos Genéticos*

AM *Aprendizado de Máquina*

DAG *Directed Acyclic Graph*

FN *Falsos Negativos*

FP *Falsos Positivos*

GC *Global Classifier*

GAD *Grafo Acíclico Direcionado*

GO *Gene Ontology*

H-Loss *Hierarchical Loss Function*

HMC-GA *Hierarchical Multi-Label Classification with a Genetic Algorithm*

HMC-LMLP *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons*

HMC-LP *Hierarchical Multi-Label Classification with Label-Powerset*

KNN *k-Nearest Neighbors*

LCL *Local Classifier per Level*

LCN *Local Classifier per Node*

LCPN *Local Classifier per Parent Node*

MIPS *Munich Information Center for Protein Sequences*

MLNP *Mandatory Leaf-Node Prediction*

MLP *Multi-Layer Perceptron*

MPP *Multiple Path Prediction*

NMLNP *Non-Mandatory Leaf-Node Prediction*

OCF *Otimização por Colônias de Formigas*

PCT *Predictive Clustering Trees*

PH *Precisão Hierárquica*

PHA *Probabilistic Hierarchical Accuracy*

PR *Precisão-Revocação*

RH *Revocação Hierárquica*

RNA *Redes Neurais Artificiais*

Rprop *Resilient Back-propagation*

SPP *Single Path Prediction*

SVM *Support Vector Machines*

VN *Verdadeiros Negativos*

VP *Verdadeiros Positivos*

Lista de Figuras

2.1	Representação de um cromossomo de um Algoritmo Genético tradicional. . . .	11
2.2	Aplicação do operador de Seleção por Roleta.	13
2.3	Aplicação do operador de Seleção por Torneio	13
2.4	Aplicação dos operadores de <i>crossover</i>	14
2.5	Aplicação do operador de mutação <i>bit-flip</i>	15
2.6	Modelo genérico de um neurônio.	17
2.7	Típico neurônio artificial.	18
2.8	Arquiteturas de redes neurais.	19
2.9	Exemplos de problemas linearmente e não linearmente separáveis.	21
2.10	Fluxo do sinal em um neurônio j	21
3.1	Problemas de classificação de artigos científicos	31
3.2	Exemplos de problemas de classificação	32
3.3	Problema de classificação hierárquica multirrótulo estruturado como um árvore	35
3.4	Representação gráfica das medidas de Precisão e Revocação Hierárquicas . . .	50
3.5	Representação gráfica da medida H-Loss	52
3.6	Exemplos de curvas Precisão-Revocação	54
3.7	Exemplo de hierarquia de classes	60
3.8	Exemplo de como a medida H-Loss pode favorecer a abordagem local.	61
4.1	Exemplo de uma hierarquia de classes de três níveis.	64
4.2	Variação HMC-LMLP-Labels.	65
4.3	Variação HMC-LMLP-Predicted.	67
4.4	Variação HMC-LMLP-True.	69
4.5	Variação HMC-LMLP-NoLabels.	70
4.6	Pós-processamento para correção de inconsistências	71
5.1	Representação de um indivíduo no método HMC-GA.	77
5.2	Exemplo de indexação de valores categóricos e operadores do método HMC-GA.	78
5.3	Exemplo do procedimento de semeadura do método HMC-GA.	79

6.1	Parte do esquema de classificação FunCat.	97
6.2	Parte do esquema de classificação da Gene Ontology	99
6.3	Comparação para a par das variações do método HMC-LMLP utilizando o teste estatístico de Nemenyi	110
6.4	Comparação para a par das variações Bp-Predicted, Rp-Predicted e métodos da literatura utilizando o teste estatístico de Nemenyi	112
6.5	Comparação para a par do método HMC-GA e métodos da literatura utilizando o teste estatístico de Nemenyi	118
6.6	Comparação para a par do método HMC-GA e métodos da literatura utilizando o teste estatístico de Nemenyi, considerando a quantidade e tamanho das regras geradas	121
6.7	Regras geradas pelos métodos HMC-GA, Clus-HMC e <i>hmAnt-Miner</i>	123
6.8	Curvas precisão-revoação obtidas nas hierarquias estruturadas como árvores	125
6.9	Curvas precisão-revoação obtidas nas hierarquias estruturadas como grafos	126

Lista de Tabelas

3.1	Categorização dos métodos de classificação hierarquia multirrótulo	48
6.1	Características dos conjuntos de dados estruturados como árvores	97
6.2	Estatísticas dos conjuntos de dados estruturados como árvores	98
6.3	Número de classes por nível nos conjuntos de dados estruturados como árvores	98
6.4	Características dos conjuntos de dados estruturados como grafos	100
6.5	Número de classes por nível nos conjuntos de dados estruturados como grafos	101
6.6	Número médio de exemplos por classe nos conjuntos de dados estruturados como grafos	102
6.7	Número médio de classes por exemplo nos conjuntos de dados estruturados como grafos	103
6.8	Resultados das quatro variações do método HMC-LMLP nos conjuntos de dados estruturados como árvores	106
6.9	Resultados das quatro variações do método HMC-LMLP nos conjuntos de dados estruturados como grafos	106
6.10	Número médio de ciclos necessários até a convergências dos algoritmos <i>Back-propagation</i> e <i>Resilient Back-propagation</i> nos conjuntos de dados estruturados como árvores	108
6.11	Número médio de ciclos necessários até a convergências dos algoritmos <i>Back-propagation</i> e <i>Resilient Back-propagation</i> nos conjuntos de dados estruturados como grafos	108
6.12	Rankings médios das variações do método HMC-LMLP de acordo com o teste de Friedman.	109
6.13	Comparação dos métodos Bp-Predicted e Rp-Predicted e métodos da literatura em conjuntos de dados estruturados como árvores	111
6.14	Comparação dos métodos Bp-Predicted e Rp-Predicted e métodos da literatura em conjuntos de dados estruturados como grafos	111
6.15	Rankings médios das variações Bp-Predicted, Rp-Predicted e métodos da literatura de acordo com o teste de Friedman.	112

6.16	Comparação par a par das variações Bp-Predicted, Rp-Predicted e métodos da literatura quanto ao número de vitórias em classes específicas das hierarquias estruturadas como grafos	113
6.17	Valores de Precisão e Revocação obtidos no conjunto de dados Celcycle estruturado como árvore	114
6.18	Parâmetros utilizados no método HMC-GA	115
6.19	Comparação dos métodos HMC-GA e métodos da literatura em conjuntos de dados estruturados como árvores	117
6.20	Comparação dos métodos HMC-GA e métodos da literatura em conjuntos de dados estruturados como grafos	117
6.21	Rankings médios do método HMC-GA e métodos da literatura de acordo com o teste de Friedman.	118
6.22	Número de regras geradas pelo método HMC-GA e métodos da literatura	119
6.23	Tamanho das regras geradas pelos métodos HMC-GA, Clus-HMC e <i>hmAnt-Miner</i>	120
6.24	Rankings médios do método HMC-GA e métodos da literatura de acordo com o teste de Friedman, considerando quantidade e tamanho das regras geradas. . .	121
6.25	Valores de $AU(\overline{PRC})$ e número/tamanho de regras obtidos em execuções específicas do método HMC-GA.	122
A.1	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados estruturados como árvores, utilizando o algoritmo <i>Back-propagation</i>	152
A.2	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados estruturados como árvores, utilizando o algoritmo <i>Resilient Back-propagation</i> .	153
A.3	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi, Eisen e Gasch1 estruturados como DAGs, utilizando o algoritmo <i>Back-propagation</i>	154
A.4	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como DAGs, utilizando o algoritmo <i>Back-propagation</i>	155
A.5	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi, Eisen e Gasch1 estruturados como DAGs, utilizando o algoritmo <i>Resilient Back-propagation</i>	156
A.6	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como DAGs, utilizando o algoritmo <i>Resilient Back-propagation</i>	157
B.1	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi, Eisen e Gasch1 estruturados como árvores, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura	160

B.2	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como árvores, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura	161
B.3	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi e Eisen estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura	162
B.4	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch1, Gasch2, Pheno e Spo estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura	163
B.5	Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Expr e Seq estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura	164
C.1	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Celcycle . . .	166
C.2	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Church . . .	167
C.3	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Derisi	168
C.4	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Eisen	169
C.5	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Gasch1 . . .	170
C.6	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Gasch2 . . .	171
C.7	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Pheno	172
C.8	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Spo	173
C.9	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Expr	174
C.10	Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Seq	175

Lista de Algoritmos

1	Algoritmo evolutivo tradicional.	11
2	Atualização de Pesos do Algoritmo Rprop.	28
3	Classificação Hierárquica Multirrótulo com um Algoritmo Genético.	76
4	<i>Crossover</i> uniforme baseado em distância.	80
5	Operador de mutação do método HMC-GA.	82
6	Operador de generalização de testes em atributos numéricos.	83
7	Operador de restrição de testes em atributos numéricos.	84
8	Operador de restrição/generalização de testes em atributos categóricos.	86
9	Operador de busca local do método HMC-GA.	87

Introdução

Classificação é uma das mais importantes tarefas da área de Aprendizado de Máquina (AM). Nessa área de pesquisa, um problema de classificação é definido como: dado um conjunto de exemplos de treinamento composto por pares $(\mathbf{x}_i, c_j)$, no qual $\mathbf{x}_i$ representa um vetor de atributos de entrada que descrevem um exemplo e c_j sua classe associada, encontrar uma função que mapeie cada $\mathbf{x}_i$ para sua classe associada c_j , tal que $i = 1, 2, \dots, n$, em que n é o número de exemplos de treinamento, e $j = 1, 2, \dots, m$, em que m é o número classes do problema.

Em problemas tradicionais de classificação, cada exemplo de um conjunto de dados é associado a apenas uma dentre duas ou mais classes. Porém, existem problemas de classificação mais complexos, nos quais cada exemplo pode ser associado a mais de uma classe simultaneamente. Adicionalmente, essas classes podem estar organizadas em uma hierarquia, que pode ser estruturada como uma árvore ou como um grafo acíclico direcionado (*Directed Acyclic Graph* (DAG)). Nesse caso, esses problemas são chamados de problemas de classificação hierárquica multirrótulo, muito comuns em tarefas como a predição de funções de genes e proteínas (Vens et al., 2008; Valentini, 2011) e classificação de documentos (Cesa-Bianchi et al., 2006; Mayne e Perry, 2009).

Problemas de classificação hierárquica multirrótulo são tratados utilizando duas abordagens principais, chamadas local ou *top-down* e global ou *one-shot* (Silla e Freitas, 2010), também encontrada em alguns trabalhos com o nome de *big-bang*. Na abordagem local, classificadores tradicionais são treinados com o objetivo de produzir uma árvore de classificadores, que são geralmente utilizados de maneira *top-down* para a classificação de um exemplo. Em contraste, a abordagem global induz um único classificador considerando a hierarquia de classes como um todo.

Na abordagem local, para a classificação de um novo exemplo, os métodos inicialmente predizem sua classe mais genérica, que é uma classe localizada no primeiro nível da estrutura hierárquica. Essa classe predita é então utilizada para reduzir o conjunto de possíveis subclas-

ses dessa classe no próximo nível. Assim, quando um exemplo é atribuído a uma classe que não seja um nó folha da hierarquia, ele é posteriormente classificado em uma ou mais subclasses dessa classe. A abordagem global, ao contrário, induz um único modelo de classificação utilizando todo o conjunto de treinamento, e a classificação de um novo exemplo ocorre em apenas um passo.

Esta pesquisa de doutorado teve como objetivo principal propor e investigar dois métodos de classificação, um utilizando a abordagem local e outro utilizando a abordagem global, para tratar problemas de classificação hierárquica multirrótulo. O restante deste capítulo está organizado da seguinte maneira: na Seção 1.1, são destacadas as motivações para a realização desta pesquisa; os métodos de classificação propostos são brevemente apresentados na Seção 1.2; a Seção 1.3 apresenta quais foram as hipóteses e objetivos principais desta pesquisa; na Seção 1.4, é apresentado um resumo dos resultados obtidos; por fim, na Seção 1.5, é apresentada a organização geral deste documento.

1.1 Motivação

A primeira motivação que levou ao desenvolvimento desta pesquisa está relacionada às abordagens global e local de classificação. Depois de uma detalhada pesquisa bibliográfica, verificou-se a inexistência de métodos de classificação hierárquica multirrótulo baseados na abordagem local que utilizem um classificador em cada nível da hierarquia. Todos os métodos encontrados, ou são baseados na abordagem global, ou associam um classificador local a cada nó da hierarquia de classes, ou associam um classificador local a cada nó interno da hierarquia de classes, decompondo o problema de classificação em vários sub-problemas. O uso de muitos classificadores pode resultar na utilização de informação muito específica, perda de informação, e não utilização das dependências entre as classes durante o processo de treinamento (Silla e Freitas, 2010). Ainda, a abordagem global deixa de utilizar informações locais que podem ser úteis para explorar diferentes padrões nos dados em diferentes níveis hierárquicos. Dessa forma, optou-se pelo desenvolvimento de um método que associe um classificador MLP a cada nível hierárquico, de maneira que os problemas previamente mencionados pudessem ser amenizados. Redes neurais foram escolhidas por serem muito robustas, e também pela facilidade de aplicação a problemas multirrótulo, já que cada neurônio de saída pode ser associado a uma classe.

A segunda motivação está relacionada à interpretabilidade dos classificadores. A tarefa de geração de regras de classificação para problemas hierárquicos multirrótulo ainda é muito desafiadora, pois uma regra deve ser capaz de classificar vários exemplos em um conjunto de classes, respeitando as restrições impostas pela hierarquia. Ainda, a interpretabilidade dessas regras está diretamente relacionada à abordagem de classificação escolhida. Métodos baseados na abordagem local geram vários conjuntos de regras, cada um produzido por um classificador. Já métodos baseados na abordagem global geram um único conjunto de regras, pois utilizam um único classificador. Assim, métodos baseados na abordagem global produzem um número

menor de regras, o que tende a ser mais interpretável do que vários conjuntos de regras.

Os principais métodos para geração de regras de classificação hierárquicas multirrótulo são baseados nas abordagens local e global utilizando Árvores de Decisão (AD) (Vens et al., 2008), ou baseados na abordagem global utilizando Otimização por Colônias de Formigas (OCF) (Otero et al., 2010). No entanto, esses métodos ainda apresentam resultados preditivos ruins. Assim, optou-se pelo desenvolvimento de um método global de classificação que utiliza Algoritmos Genéticos (AG) para a geração de regras de classificação. AG foram escolhidos por se tratarem de algoritmos bem estabelecidos na literatura para a geração de regras, e robustos para muitos problemas. Ainda, a função objetivo nos AG não é diferenciável. Utilizou-se a busca global realizada pelo algoritmo genético para gerar regras de classificação simples e com bom desempenho preditivo.

A terceira motivação está relacionada às áreas de aplicação de métodos hierárquicos multirrótulo. Uma dessas áreas é a Bioinformática, em que técnicas computacionais têm muito a contribuir para a área da proteômica, por meio da identificação de proteínas expressas pelo genoma e predição de suas funções. Essa predição pode ser feita por meio de homologia, comparando uma nova sequência com outras sequências de proteínas, já com funções identificadas, presentes em um conjunto de dados. Apesar desse método ser muito utilizado, ele apresenta limitações. Duas proteínas podem apresentar sequências similares mas desempenharem funções diferentes, ou apresentar sequências diferentes e desempenharem funções iguais ou similares. Proteínas sendo comparadas podem ser similares em regiões de suas sequências não determinantes para suas funções. Adicionalmente, a predição por meio de homologia entre sequências ignora as muitas propriedades bioquímicas das proteínas. Muitas bases de dados são organizadas de maneira hierárquica, quando são distribuídas em classes e subclasses, e alguns problemas hierárquicos são multirrótulo (proteínas podem desempenhar mais de uma função), dificultando ainda mais a tarefa de classificação. Sendo assim, métodos de predição baseados em AM têm sido explorados na classificação de funções de proteínas. Nesta pesquisa, optou-se por trabalhar com problemas de predição de funções de proteínas, nos quais as funções estão organizadas em estruturas do tipo árvore e do tipo Grafo Acíclico Direcionado (GAD) (do inglês *Directed Acyclic Graph* (DAG)).

Com hierarquias estruturadas como árvores, foram utilizados conjuntos de dados com classes organizadas de acordo com o esquema de classificação FunCat (Ruepp et al., 2004), desenvolvido pelo *Munich Information Center for Protein Sequences* (MIPS). Nesse esquema, as classes (funções de proteínas) são identificadas por um código composto por números, sendo que o primeiro especifica a classe mais geral e o último a classe mais específica. Exemplificando, a função *metabolismo da glutamina* tem o código 01.01.03, em que 01 representa a função *metabolismo*, 01.01 representa a função *metabolismo de aminoácido*, e 01.01.03 representa a função *assimilação de amônia (metabolismo do grupo glutamato)*. A hierarquia de classes da FunCat possui seis níveis hierárquicos e centenas de classes.

Estruturados como grafos, foram utilizados conjuntos de dados com classes organizadas de acordo com a *Gene Ontology* (GO) (Ashburner et al., 2000). Essa hierarquia é organizada

em três ontologias: ontologia de componentes celulares, ontologia de processos biológicos e ontologia de funções moleculares. Nessas ontologias, funções de proteínas estão organizadas de maneira hierárquica, e proteínas podem apresentar mais de uma função. Os conjuntos de dados utilizados possuem milhares de classes e podem possuir até 13 níveis hierárquicos, dependendo do caminho escolhido entre uma classe e o nó raiz da hierarquia.

1.2 Métodos Propostos

Os dois métodos propostos e investigados nesta pesquisa foram chamados de *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* ([HMC-LMLP](#)) e *Hierarchical Multi-Label Classification with a Genetic Algorithm* ([HMC-GA](#)). Esses métodos, brevemente apresentados a seguir, serão explicados detalhadamente nos Capítulos 4 e 5, respectivamente.

O princípio por trás do método [HMC-LMLP](#) é associar uma rede neural [MLP](#) a cada nível da hierarquia de classes. Assim, cada rede neural é responsável pelas predições no seu nível associado. Cada rede neural possui o número de neurônios de saída igual ao número de classes de seu nível associado, o que faz com que uma rede possa prever mais de uma classe simultaneamente. Redes neurais foram utilizadas por serem classificadores robustos, e também porque deu-se prioridade ao desempenho preditivo do método, e não a sua interpretabilidade. Ainda, a associação de cada neurônio da camada de saída a uma classe facilita a aplicação das redes neurais em problemas multirrotulo.

A característica mais importante do método [HMC-LMLP](#) é que a saída de uma rede neural associada ao nível l é utilizada para complementar os vetores de atributos dos exemplos utilizados no treinamento da rede neural associada ao nível $l + 1$. Assim, as redes [MLP](#) são treinadas incrementalmente. Como exemplo, considere uma hierarquia de três níveis. Primeiramente, a rede neural associada ao primeiro nível é treinada utilizando todos os exemplos de treinamento. Quando esse treinamento é finalizado, o treinamento da rede neural associada ao segundo nível é iniciado. Esse treinamento ocorre da seguinte maneira. Um exemplo é dado como entrada para a primeira rede neural, previamente treinada, e tem suas predições registradas. Se esse exemplo pertence ao segundo nível, a saída da rede neural anterior (predições do primeiro nível) é utilizada para complementar o vetor de atributos do exemplo, que é então fornecido como entrada para a rede neural associada ao segundo nível. Esse processo é repetido para todos os exemplos e até que o processo de treinamento dessa rede seja finalizado. Para treinar a rede neural do terceiro nível, o mesmo processo é repetido, mas agora considerando as duas redes neurais anteriores. Inicialmente, um exemplo é fornecido como entrada para a rede neural do primeiro nível e tem seu vetor de atributos complementado pela saída dessa rede. O exemplo, se pertencente ao nível 2, é então fornecido como entrada para a rede neural do segundo nível. Se esse exemplo pertence ao terceiro nível, a saída da rede do nível 2 é utilizada para complementar o vetor de atributos original do exemplo, que então é fornecido como entrada para a rede neural associada ao terceiro nível.

Além de tentar amenizar os problemas relacionados às abordagens global e local, como

mencionado na seção anterior, o método **HMC-LMLP** tenta fazer com que a rede neural associada a um nível possa aprender algo diferente da rede neural associada ao nível anterior. Para isso, cada rede utiliza informação que foi aprendida pela rede neural previamente treinada (predições do nível anterior). Essa informação é fornecida às redes por meio da complementação dos vetores de atributos dos exemplos utilizados no treinamento. Além disso, espera-se que essa complementação faça com que as redes **MLP** aprendam elas mesmas os relacionamentos existentes entre as classes da hierarquia.

O método **HMC-GA** utiliza um procedimento de cobertura sequencial de exemplos para a geração de regras ao mesmo tempo hierárquicas e multirrótulo. Cada execução desse procedimento gera um conjunto de regras. A melhor regra é então salva e os exemplos de treinamento cobertos por ela são retirados do conjunto de treinamento. O procedimento de cobertura sequencial é então reiniciado com os exemplos restantes do conjunto de dados, e é repetido até que todos ou quase todos os exemplos (parâmetro fornecido pelo usuário) sejam cobertos. Os operadores genéticos implementados foram utilizados para a geração dos antecedentes das regras. Os consequentes foram gerados por meio de um procedimento determinístico que utiliza as classes dos exemplos cobertos pelos antecedentes das regras geradas. O consequente de uma regra é dado por um vetor, no qual cada posição corresponde a uma classe. Cada posição então recebe um valor real, que representa a proporção de exemplos cobertos pela regra, e que pertencem à classe naquela posição.

Diferentemente do método **HMC-LMLP**, buscou-se aqui a obtenção de um classificador que fosse ao mesmo tempo interpretável (conjunto de regras) e obtivesse desempenho de classificação competitivo com os métodos estado-da-arte da literatura. O algoritmo genético desenvolvido possui um operador de *crossover* que busca cruzar regras que cobrem exemplos que sejam classificados em um conjunto similar de classes, de maneira a especializar as regras geradas. Operações de mutação também são utilizadas para generalizar ou restringir regras, e uma função de *fitness* que valoriza regras que cobrem um conjunto mais homogêneo de exemplos também foi utilizada.

1.3 Hipótese e Objetivo

Como mencionado anteriormente, o objetivo principal desta pesquisa foi propor e investigar dois métodos de classificação. O primeiro utiliza redes neurais e é baseado na abordagem local de classificação. O segundo utiliza algoritmos genéticos para a geração de regras de classificação e é baseado na abordagem global. O objetivo do método local é simplesmente obter bons resultados preditivos, enquanto que com o método global busca-se um modelo interpretável que ainda assim consiga resultados preditivos competitivos com os métodos da literatura. Os métodos foram aplicados à tarefa de predição de funções de proteínas. Assim, trabalha-se com duas hipóteses, listadas a seguir.

- A utilização de uma rede neural local por nível, e também a complementação dos vetores de atributos dos exemplos com as predições, leva à melhora do desempenho preditivo,

fazendo com que cada rede neural em um nível aprenda algo diferente da rede neural do nível anterior, e também fazendo com que os relacionamentos entre as classes sejam aprendidos pelas redes durante o processo de treinamento;

- Um algoritmo genético utilizando um procedimento de cobertura sequencial de exemplos, e baseado na abordagem global, é capaz de gerar regras de classificação hierárquicas multirrótulo que sejam simples e com desempenho preditivo competitivo.

Formuladas as hipótese desta pesquisa, o objetivo principal foi concretizado, em linhas gerais, por meio do cumprimento das seguintes etapas:

- **Implementação e análise de novos métodos:** após o estudo dos principais métodos propostos na literatura, foi iniciada a fase de implementação de novos métodos de classificação utilizando redes neurais e algoritmos genéticos. Os métodos propostos foram então comparados com outros métodos da literatura, inclusive com um método considerado estado-da-arte;
- **Aplicação dos métodos propostos em problemas relevantes:** a área da Bioinformática, especificamente a tarefa de predição de funções de proteínas, foi escolhida como foco dos experimentos. Essa escolha deveu-se ao fato da existência de conjuntos de dados estruturados em diferentes taxonomias, árvore e grafo. Além disso, é um problema muito interessante do ponto de vista biológico, pois saber a função de uma proteína é fundamental, entre outras razões, para entender seu papel no desenvolvimento e evolução dos seres vivos. A tarefa de predição de funções de proteínas possui uma grande bibliografia, e também conjuntos de dados públicos que foram utilizados por diferentes métodos de classificação da literatura, facilitando a comparação e reprodução de resultados;
- **Avaliação e validação dos resultados obtidos:** levando em consideração as características peculiares de problemas de classificação hierárquica multirrótulo, diferentes medidas de avaliação específicas foram desenvolvidas para tais problemas. Após estudos envolvendo essas medidas, uma medida específica foi escolhida para avaliar os métodos propostos, por se tratar da mais abrangente, eficaz e de fácil implementação. Testes estatísticos também foram realizados para determinar se houve diferença estatística de desempenho entre os métodos comparados;
- **Revisão bibliográfica:** durante todo o decorrer da pesquisa de Doutorado, foi realizado um amplo levantamento bibliográfico de trabalhos relacionados. Com esse levantamento, foi possível escolher os métodos da literatura que foram comparados com os métodos propostos nesta pesquisa. Por meio da revisão bibliográfica, também foram escolhidas as medidas de avaliação, e selecionados os conjuntos de dados utilizados nos experimentos.

1.4 *Resumo dos Resultados*

Os experimentos desta Tese foram divididos em duas etapas. Primeiramente, foram analisados os resultados obtidos com o método [HMC-LMLP](#) e, posteriormente, os resultados obtidos com o método [HMC-GA](#).

Considerando o método [HMC-LMLP](#), experimentos foram realizados para verificar se a utilização das predições para complementar os vetores de atributos dos exemplos levou à melhora no desempenho de classificação. Para isso, diferentes variações do método foram propostas, utilizando e não utilizando a complementação dos vetores de atributos. Verificou-se que a utilização das predições melhorou o desempenho de classificação, principalmente nas hierarquias estruturadas como árvores.

Os resultados obtidos com o método [HMC-LMLP](#) podem ser considerados muito bons, levando em consideração as dificuldades do problema de classificação. Em hierarquias estruturadas como árvores, as redes neurais obtiveram desempenhos de classificação superiores a todos os métodos da literatura utilizados nos experimentos. Ainda, em hierarquias estruturadas como grafos, os desempenhos foram muito satisfatórios, já que foram competitivos e, em alguns casos, superiores aos métodos da literatura.

Os experimentos com o método [HMC-GA](#) mostraram que, no geral, não foram obtidos desempenhos de classificação melhores que o método considerado estado-da-arte na literatura. No entanto, o método conseguiu resultados melhores ou competitivos na comparação com outros métodos da literatura, baseados tanto nas abordagens global quanto local.

Apesar de não superar o desempenho do método estado-da-arte, os resultados obtidos com o método [HMC-GA](#) podem ser considerados animadores, já que em alguns casos, desempenhos competitivos foram obtidos com a indução de uma quantidade menor de regras. Ainda nesses casos, foram geradas regras menores do que as regras geradas pelos outros métodos.

1.5 *Organização do Documento*

O restante deste documento está organizado da seguinte maneira: o Capítulo [2](#) apresenta os conceitos fundamentais de redes neurais e computação evolutiva, necessários para a realização desta pesquisa; os conceitos fundamentais de classificação hierárquica e multirrótulo são apresentados no Capítulo [3](#), juntamente com uma revisão bibliográfica de trabalhos que abordam o tema e também uma revisão sobre medidas de avaliação utilizadas em problemas hierárquicos multirrótulo; o Capítulo [4](#) apresenta detalhadamente o método [HMC-LMLP](#); o método [HMC-GA](#) é apresentado em detalhes no Capítulo [5](#); todos os experimentos e as análises realizadas são apresentados no Capítulo [6](#); por fim, o Capítulo [7](#) apresenta as conclusões dessa pesquisa e apontamentos para trabalhos futuros.

Conceitos Fundamentais de Redes Neurais e Computação Evolutiva

Este capítulo apresenta os conceitos fundamentais relacionados a Redes Neurais e Algoritmos Evolutivos, áreas nas quais se inserem os dois métodos propostos nessa pesquisa de Doutorado. O restante do capítulo está organizado da seguinte maneira: a Seção 2.1 apresenta os conceitos de Computação Evolutiva, tratando especificamente de Algoritmos Genéticos; a Seção 2.2 apresenta os conceitos de Redes Neurais Artificiais, com ênfase nos algoritmos de treinamento *Back-propagation* e *Resilient Back-propagation*; por fim, na Seção 2.3 são feitas as considerações finais do capítulo.

2.1 Computação Evolutiva

A Computação Evolutiva (CE) utiliza ideias da biologia evolutiva para desenvolver métodos de busca e otimização para a resolução de problemas complexos (de Castro, 2006). A maioria dos algoritmos desenvolvidos nessa área é baseada na teoria da evolução dos seres vivos descrita por Charles Darwin em seu livro “A Origem das Espécies” (Darwin, 1872). A teoria proposta por Darwin diz que se uma população de indivíduos for capaz de se reproduzir e estiver sujeita a uma herança com variação genética, seguida por um processo de seleção, novas populações desses indivíduos surgirão, com uma tendência de se apresentarem melhor adaptadas ao seu ambiente. De acordo com essa teoria, os indivíduos de uma população que são mais adaptados têm maior chance de sobrevivência. Os algoritmos que pertencem a esse paradigma possuem características como comportamento adaptativo (Goldberg e Holland, 1988).

O termo CE é utilizado para descrever a área de pesquisa a que pertencem todos os tipos de algoritmos evolutivos. Todos esses algoritmos fazem uso do processo de evolução natural como um paradigma para resolução de problemas. A CE pode ser dividida em quatro grupos

principais de algoritmos: Algoritmos Genéticos (AG) (Holland, 1975), Estratégias Evolutivas (EE) (Beyer e Schwefel, 2002), Programação Evolutiva (PE) (Bäck et al., 2000a,b) e Programação Genética (PG) (Koza, 1992). Apesar dessas abordagens serem diferentes, todos os seus algoritmos possuem as mesmas características básicas de um processo evolutivo, como proposto na teoria da evolução de Darwin. Um algoritmo evolutivo tradicional possui as seguintes características (de Castro, 2006):

- **Codificação de soluções:** soluções candidatas (indivíduos) são codificadas, criando o conceito de espaço de busca;
- **Uma população de indivíduos que se reproduzem com herança:** cada indivíduo representa ou codifica um ponto em um espaço de busca de potenciais soluções para um dado problema. Esses indivíduos podem se reproduzir e gerar descendentes que herdaram características de seus pais;
- **Variação genética:** os indivíduos são susceptíveis à variação genética por meio de mutações, que alteram suas estruturas. As mutações permitem o aparecimento de novas características em uma população e, portanto, permitem a exploração de novas regiões no espaço de busca;
- **Seleção natural:** a avaliação dos indivíduos em seu ambiente é feita por uma medida de adaptabilidade, qualidade, ou valor de *fitness* atribuído a cada indivíduo. A competição entre os indivíduos pela sobrevivência e recombinação é feita por meio da comparação de seus valores de *fitness*, sendo que os indivíduos com maior valor de *fitness* possuem vantagem seletiva.

2.1.1 Algoritmo Evolutivo Tradicional

O algoritmo evolutivo tradicional é genérico, iterativo e probabilístico. Ele é iniciado com uma população $\mathcal{P}$ de N indivíduos, com $\mathcal{P} = \{\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_N\}$, e a cada iteração t os indivíduos são avaliados quanto aos seus respectivos valores de *fitness* (f). Na iteração $t + 1$, uma nova população é gerada a partir da seleção e recombinação dos indivíduos. Esse cruzamento de indivíduos pode ser feito por meio de recombinação genética (*crossover*), e variações genéticas podem ser obtidas por meio de mutações em alguns indivíduos da população. A recombinação genética e a mutação ocorrem de acordo com probabilidades específicas para essas duas operações, p_c e p_m , respectivamente. Uma geração é obtida quando são finalizadas as etapas de recombinação, mutação e seleção dos melhores indivíduos.

A maioria dos algoritmos evolutivos pode ser implementada utilizando o procedimento apresentado no Algoritmo 1, com algumas diferenças nos operadores de seleção, recombinação e mutação, e também na ordem em que esses operadores são aplicados. Como critério de parada, geralmente é utilizado um número máximo de gerações ou o alcance de algum objetivo previamente especificado.

Algoritmo 1: Algoritmo evolutivo tradicional (Eiben e Smith, 2003).

procedimento AE-padrão($p_c, p_m, \mathcal{P}$)

Entrada: probabilidade de *crossover* p_c
 probabilidade de mutação p_m
 população $\mathcal{P}$

Saída: população $\mathcal{P}$ evoluída

```

1 inicialização( $\mathcal{P}$ )
2  $f \leftarrow$  avaliação( $\mathcal{P}$ )
3  $t \leftarrow 0$ 
4 enquanto critério de parada não for satisfeito faça
5    $\mathcal{P} \leftarrow$  seleção( $\mathcal{P}, f$ )
6    $\mathcal{P} \leftarrow$  recombinação( $\mathcal{P}, p_c$ )
7    $\mathcal{P} \leftarrow$  mutação( $\mathcal{P}, p_m$ )
8    $f \leftarrow$  avaliação( $\mathcal{P}$ )
9    $t \leftarrow t + 1$ 

```

A seguir, serão apresentados os conceitos fundamentais dos AGs, abordagem evolutiva investigada nesta pesquisa de doutorado.

2.1.2 Algoritmos Genéticos

Os AGs possuem todos os processos básicos da teoria de Darwin presentes no Algoritmo 1, porém os termos utilizados nos AGs são termos utilizados em Genética. Entre as várias versões de AGs utilizadas, a mais comum é a apresentada por Holland (1975).

Os indivíduos de um Algoritmo Genético são codificados em genótipos, e são representados por estruturas de dados chamadas de cromossomos. Os genótipos podem ser codificados de várias maneiras, como por exemplo de forma binária, inteira ou ponto flutuante. Neste capítulo, para exemplificação dos conceitos envolvidos em AGs, será utilizada a codificação binária. Nessa codificação, os genótipos são representados por *strings* binárias, contendo os valores 0 ou 1, e cada unidade de um cromossomo é chamada de gene, sendo que cada gene é localizado em um local específico do cromossomo, chamado *locus*. As diferentes formas, ou valores, que um gene pode assumir, são chamadas de alelos. A Figura 2.1 ilustra uma *string* binária de tamanho 10, representando um cromossomo de um Algoritmo Genético.

Figura 2.1: Representação de um cromossomo de um Algoritmo Genético tradicional. Nesse caso, cada posição (*locus*) do cromossomo pode assumir um de dois possíveis alelos, 0 ou 1. Adaptado de (de Castro, 2006).

Para a resolução de um problema, deve ser definida uma função objetivo que permita a avaliação do *fitness* de cada potencial solução do problema, representada por um genótipo, ou seja, um cromossomo. O significado de cada genótipo é o seu fenótipo, e também deve ser definido

de acordo com o problema a ser solucionado. Um Algoritmo Genético, portanto, é executado em uma população de cromossomos, e faz uma busca por todo o espaço de potenciais soluções.

Durante a execução de um Algoritmo Genético, cada cromossomo da população é avaliado (procedimento “avaliação” do Algoritmo 1) de acordo com a função objetivo, e tem um valor de *fitness* associado. A seleção dos melhores cromossomos (procedimento “seleção” do Algoritmo 1) para recombinação, pode ser feita por um operador de seleção chamado Seleção por Roleta (Fogel, 2000). Nesse caso, a probabilidade de cada cromossomo ser selecionado para recombinação é proporcional ao seu *fitness*, ou seja, quanto maior o *fitness* de um cromossomo, maior a probabilidade de ele ser selecionado para recombinação, e vice-versa.

A Seleção por Roleta, por ser um método de seleção proporcional ao *fitness*, pode fazer com que o algoritmo tenha uma convergência prematura. Isso acontece quando os valores dos *fitness* dos indivíduos são muito discrepantes (poucos indivíduos com valores de *fitness* muito altos, e muitos indivíduos com valores de *fitness* muito baixos). Nesse caso, como esses poucos indivíduos possuem uma probabilidade muito maior de serem selecionados, a população tenderá a convergir rapidamente para apenas descendentes desses indivíduos, possivelmente produzindo soluções subótimas.

Outro método de seleção de indivíduos é a Seleção por Torneio (Wetzel, 1983). Nesse método, $n \ll N$ indivíduos são selecionados aleatoriamente da população, e então o indivíduo com maior valor de *fitness* é selecionado. O método de Seleção por Torneio ameniza o problema da convergência prematura, pois todos os indivíduos têm a mesma chance de serem selecionados para participar do torneio.

Após o procedimento de seleção dos melhores cromossomos de acordo com seus valores de *fitness*, os cromossomos selecionados dão origem a descendentes, por meio de operadores genéticos específicos chamados de *crossover* e mutação. O operador de *crossover* tem a função de fazer a recombinação genética dos cromossomos selecionados (procedimento “recombinação” do Algoritmo 1). A mutação é um método de variação genética, e é representada pelo procedimento “mutação” do Algoritmo 1. Os procedimentos de Seleção por Roleta, Seleção por Torneio, *crossover* e mutação são brevemente apresentados a seguir.

Seleção por Roleta

No método de Seleção por Roleta (Holland, 1975; Goldberg, 1989), como o próprio nome sugere, cada cromossomo é atribuído a uma partição de uma roleta, sendo que cada partição tem tamanho proporcional ao valor do *fitness* do cromossomo atribuído a ela. Assim, boas soluções são atribuídas a partições maiores na roleta, tendo uma probabilidade maior de serem selecionadas quando a roleta é girada para a seleção de cromossomos para recombinação. A Seleção por Roleta pode ser implementada da seguinte maneira (Sastry et al., 2005):

1. avaliação do *fitness* f_i de cada cromossomo da população;
2. cálculo da probabilidade p_i (tamanho da partição) da seleção de cada cromossomo da população: $p_i = f_i / \sum_{j=1}^N f_j$, sendo N o tamanho da população;

3. cálculo da probabilidade cumulativa, q_i , de cada cromossomo: $q_i = \sum_{j=1}^i p_j$;
4. geração de um número aleatório uniforme, $r \in (0, 1]$;
5. se $r < q_1$, seleccionar o primeiro cromossomo, $\mathbf{x}_1$, senão, seleccionar o cromossomo $\mathbf{x}_i$ tal que $q_{i-1} < r \leq q_i$;
6. Repetir os passos 4 e 5 n vezes, para seleccionar n candidatos para recombinação.

A Figura 2.2 ilustra o operador de Seleção por Roleta aplicado em uma população composta por quatro cromossomos. Na figura, a roleta é dividida em 24 seções de 15 graus cada, e cada cromossomo é atribuído a uma partição formada pelo número de seções correspondente ao valor de seu *fitness*.

Figura 2.2: Aplicação do operador de Seleção por Roleta. Adaptado de (de Castro, 2006).

Seleção por Torneio

No método de Seleção por Torneio (Wetzel, 1983), cromossomos são seleccionados aleatoriamente para participarem de um torneio, no qual competirão entre si para que seja decidido qual deles participará do processo de recombinação. O processo de seleção para a participação no torneio pode ocorrer com ou sem reposição, e vence o torneio o cromossomo que tiver o maior valor de *fitness*. Assim, são necessários n torneios para que sejam escolhidos n indivíduos para recombinação. A Figura 2.3 ilustra o operador de Seleção por Torneio.

Figura 2.3: Aplicação do operador de Seleção por Torneio, com tamanho do torneio igual a três.

Crossover de Acordo com a Codificação Binária

O operador de *crossover* é utilizado para recombinar os cromossomos selecionados, e assim criar novos, e esperados melhores, descendentes. Vários operadores de *crossover* podem ser encontrados na literatura (Goldberg, 1989; Booker et al., 1997; Spears, 1997). Serão apresentados aqui dois operadores de *crossover* muito utilizados, chamados de *crossover* de k pontos e *crossover* uniforme.

Na maioria dos operadores de *crossover*, os indivíduos selecionados são recombinados com uma probabilidade p_c , ou seja, é gerado um número aleatório uniforme $r \in [0, 1]$, e se $r \leq p_c$, os dois cromossomos são recombinados, senão os descendentes gerados são cópias de seus pais. Os operadores de *crossover* uniforme e de k pontos podem ser definidos da seguinte maneira:

- **Crossover de k pontos:** são escolhidas aleatoriamente k regiões dos cromossomos, que são então divididos nessas regiões e têm seus alelos trocados entre si. Os operadores mais simples e mais utilizados são o de um ponto e o de dois pontos. As Figuras 2.4(a) e 2.4(b) ilustram a aplicação dos operadores de *crossover* de um e dois pontos, respectivamente;
- **Crossover uniforme:** todos os alelos são trocados entre dois cromossomos com uma probabilidade p_t , chamada probabilidade de troca. Geralmente é escolhido o valor 0.5 para essa probabilidade. A Figura 2.4(c) ilustra a aplicação do operador de *crossover* uniforme.

Figura 2.4: Aplicação dos operadores de *crossover*: (a) um ponto, (b) dois pontos e (c) uniforme. Adaptado de (Sastry et al., 2005).

Mutação de Acordo com a Codificação Binária

O operador de mutação tem o objetivo de aleatoriamente modificar um alelo de um gene, ou então substituí-lo por outro, de maneira que seja criado um novo cromossomo. Essa modificação é feita para adicionar diversidade à população e garantir que potencialmente todo o espaço de busca possa ser explorado. Em AGs, o operador de mutação é aplicado após o operador de *crossover*, de acordo com uma probabilidade baixa de mutação (p_m) para cada gene (Sastry et al., 2005). Um dos operadores de mutação mais utilizados é o que troca um *bit* por outro (*bit-flip*) em um cromossomo representado por uma *string* binária, ou seja, o cromossomo é percorrido e cada alelo é trocado (0 trocado por 1 e vice-versa) com uma certa probabilidade p_m . Além desse operador simples, existem também outros operadores desenvolvidos especificamente para determinados problemas. A Figura 2.5 ilustra a aplicação do operador de mutação *bit-flip*.

Figura 2.5: Aplicação do operador de mutação *bit-flip*. Adaptado de (de Castro, 2006).

2.2 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) podem ser definidas como sistemas computacionais que processam informações baseados no funcionamento do cérebro e nas conexões de seus neurônios, com o objetivo de solucionar problemas. Em (Haykin, 1999), uma Rede Neural Artificial é definida como um processador distribuído paralelo composto de unidades de processamento simples, que possui uma propensão natural para armazenar conhecimento experimental e torná-lo disponível para utilização. Ainda, segundo Haykin (1999), o conhecimento é adquirido pela rede a partir de seu ambiente, por meio de um processo de aprendizado, e as forças das conexões entre os neurônios artificiais da rede, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido. Outras definições para as RNAs também podem ser encontradas na literatura:

- “Uma Rede Neural Artificial é um circuito composto de um número muito grande de elementos de processamento simples que são baseados em neurônios” (Nigrin, 1993).
- “... a computação baseada em neurônios (*neurocomputing*) é uma disciplina tecnológica que envolve sistemas processadores de informações paralelos, distribuídos e adaptativos, que desenvolvem capacidades de processar informações em resposta à exposição a um ambiente com informações. As estruturas de processamento de informações de principal interesse nessa disciplina são as redes neurais ...” (Hecht-Nielsen, 1989).

Esta seção apresenta os conceitos fundamentais das **RNAs**, focando nas arquiteturas mais comuns, de uma camada e de múltiplas camadas.

2.2.1 O Sistema Nervoso

O sistema nervoso dos organismos desempenha o papel de processar as informações recebidas (sinais) e fornecer reações apropriadas de acordo com esses sinais. Até mesmo organismos mais simples, como lesmas e insetos, possuem a habilidade de aprender e armazenar informações em seu sistema nervoso. As redes neurais são formadas pela interconexão de muitas unidades de processamento elementares do sistema nervoso, chamadas de neurônios ou células nervosas, especializadas em processamento de sinais. No cérebro humano, por exemplo, cada neurônio possui centenas ou milhares de conexões com outros neurônios (de Castro, 2006). Os neurônios são capazes de gerar um sinal, mais especificamente um potencial elétrico, que é utilizado para transmitir informações a outros neurônios conectados. Os neurônios são conectados por meio de regiões chamadas sinapses, importantes para o entendimento do processamento de sinais no sistema nervoso (Trappenberg, 2002).

O sistema nervoso de um organismo possui duas divisões principais, chamadas Sistema Nervoso Central e Sistema Nervoso Periférico, localizadas em regiões diferentes. Nos animais vertebrados, o Sistema Nervoso Central é localizado no crânio, e o Sistema Nervoso Periférico se estende pelo resto do corpo. A parte do sistema nervoso localizada no crânio, o cérebro, pode ser dividida em três estruturas principais, chamadas cérebro anterior, tronco encefálico e cerebelo. O cérebro anterior pode ser dividido em telencéfalo e diencéfalo. O telencéfalo ocupa a maior parte do cérebro, e recebe todos os sinais que vêm do corpo pelo tronco encefálico e diencéfalo. No telencéfalo, o córtex cerebral processa os sinais recebidos, além de seus próprios sinais internos. No diencéfalo estão localizados o tálamo e o hipotálamo. O tálamo é um conjunto de núcleos por onde passam quase todas as informações encaminhadas ao córtex cerebral, que corresponde à camada mais externa do cérebro, muito rica em neurônios, e responsável por funções como o pensamento, movimentos voluntários, linguagem, raciocínio e percepção. O hipotálamo possui funções relativas ao comportamento do corpo, como fome, sede, sono, alerta, liberação de hormônios e outras funções biológicas críticas.

Pelo tronco encefálico passam todas as fibras nervosas que transportam os sinais de entrada e saída, conectando o cérebro à medula espinhal. No tronco encefálico, estão localizados o mesencéfalo, a ponte e o bulbo, que desempenham funções sensoriais e motoras, como movimentos dos olhos e controle dos músculos da cabeça. O cerebelo representa um volume pequeno de todo o cérebro, porém possui uma massa grande com uma grande quantidade de neurônios, e é responsável por monitorar e ajustar, em tempo real, o funcionamento do córtex cerebral.

2.2.2 Neurônios e Sinapses

A capacidade que o cérebro tem de processar informações está relacionada com a transmissão de sinais entre seus neurônios. A modulação dos sinais transmitidos entre os neurônios

permite que o cérebro adapte-se a diferentes situações. Acredita-se que isso seja a base para associações, memorização e muitas outras habilidades mentais (Trappenberg, 2002).

A Figura 2.6 ilustra um modelo genérico de um neurônio. O neurônio biológico é uma única célula composta, simplificada, por um corpo celular, dendritos e um axônio. Os dendritos recebem sinais de outros neurônios, enquanto o axônio propaga o sinal de saída para outros neurônios. Os neurônios que enviam sinais são chamados pré-sinápticos e os neurônios que recebem sinais são chamados pós-sinápticos. A junção entre os neurônios pré e pós-sinápticos acontece nas sinapses, nas terminações do axônio.

A característica que as sinapses possuem de processar informações permite que elas alterem o estado de um neurônio pós-sináptico por meio de um impulso elétrico, que viaja pelo axônio do neurônio e alcança outro neurônio, permitindo que informações sejam levadas a diferentes regiões do sistema nervoso. Assim, um neurônio pode ser visto como um dispositivo capaz de receber diversos estímulos de entrada de muitos neurônios e propagar um único sinal de saída para vários outros neurônios (de Castro, 2006).

Figura 2.6: Modelo genérico de um neurônio.

No cérebro, vários neurônios são interconectados formando redes neurais com capacidades complexas de comportamento e armazenamento de informações. Em algumas áreas do cérebro, os neurônios também são organizados em camadas, sendo que camadas diferentes recebem impulsos diferentes, como por exemplo impulsos visuais e auditivos.

2.2.3 Neurônios Artificiais

Quando neurônios artificiais são organizados em rede para formar uma Rede Neural Artificial, essa rede possui algumas características em comum com as redes neurais biológicas (de Castro, 2006):

- O processamento das informações ocorre em unidades simples chamadas neurônios artificiais;
- Esses neurônios são capazes de receber e enviar sinais de outros neurônios;
- Neurônios podem ser conectados a outros neurônios formando redes neurais;
- Informações (sinais) são transmitidos entre os neurônios através de sinapses;
- A eficiência de uma sinapse, representada pelo seu peso associado, corresponde à informação armazenada no neurônio, portanto, na rede neural;

- O conhecimento é adquirido do ambiente por meio de um processo de aprendizado, que é basicamente a adaptação dos pesos das conexões aos estímulos do ambiente.

Uma das primeiras tentativas de entender a atividade cerebral utilizando unidades de processamento simples de computação (neurônios artificiais) foi feita por McCulloch e Pitts (1943). O neurônio proposto por McCulloch e Pitts recebe sinais de entrada provenientes de sinapses j associadas a pesos w_j . Seu funcionamento é muito simples. A cada iteração t , o neurônio responde a suas entradas sinápticas, que refletem o estado dos neurônios pré-sinápticos. O neurônio então soma suas entradas sinápticas (valores das sinapses de entrada multiplicados pelos seus respectivos valores de peso associados), e verifica se essa soma é maior ou igual a 0, utilizando uma função de ativação chamada função degrau ou limiar. Se a soma for maior ou igual a 0, o neurônio é ativado e tem como saída o valor 1. Caso contrário, o neurônio permanece inativo, tendo como saída o valor 0.

A eficiência das sinapses de entrada individuais pode ser melhor levada em consideração se forem utilizados pesos w_j diferentes para cada sinapse j . Assim, a entrada de um neurônio no tempo é dada pela soma dos valores das sinapses de entrada multiplicados pelos seus respectivos valores de peso. A Figura 2.7 ilustra um típico neurônio artificial com suas partes mais importantes: as sinapses, caracterizadas por seus valores de peso associados a cada entrada do neurônio, uma função somadora, e uma função de ativação. A função de ativação é utilizada para limitar a amplitude do sinal de saída do neurônio a algum valor finito (Rumelhart e McClelland, 1986). Além da função degrau, várias funções de ativação podem ser utilizadas, como a função linear e a função sigmoidal.

Figura 2.7: Típico neurônio artificial. Adaptado de (Haykin, 1999).

O bias b_k da Figura 2.7 tem o efeito de aumentar ou diminuir o valor de entrada da função de ativação, dependendo se esse valor for positivo ou negativo, respectivamente (Haykin, 1999). Matematicamente, o neurônio da Figura 2.7 pode ser representado pela Equação (2.1).

$$y_k = f(u_k) = f\left(\sum_{j=1}^m w_{kj}x_j + b_k\right) \quad (2.1)$$

Na Equação 2.1, x_j , $j = 1, \dots, m$, são os sinais de entrada; w_{kj} , $j = 1, \dots, m$, são os pesos sinápticos do neurônio k ; u_k é a entrada da rede para a função de ativação; b_k é o bias do neurônio

k ; $f(\cdot)$ é a função de ativação; e y_k é o sinal de saída do neurônio. A notação dessa equação pode ser simplificada se o bias for definido como um sinal de entrada constante $x_0 = 1$ conectado ao neurônio k com um valor de peso $w_{k0} = b_k$ associado (Equação (2.2)).

$$y_k = f(u_k) = f\left(\sum_{j=0}^m w_{kj}x_j\right) \quad (2.2)$$

Como já mencionado, neurônios individuais podem ser interconectados para formar estruturas mais complexas. No cérebro humano, vários neurônios interconectados desempenham várias funções, afetando o estado de outros neurônios. Apesar de não se saber muito sobre como os neurônios do cérebro humano são interconectados, têm-se algum conhecimento sobre regiões específicas do cérebro. No córtex cerebral, por exemplo, podem ser definidas organizações neurais em termos de camadas de neurônios. Nas **RNA**, as camadas de neurônios são camadas funcionais de nós, organizadas de maneira a solucionar problemas específicos. De maneira geral, existem três tipos de arquiteturas de **RNA**: redes neurais de uma camada, redes neurais de múltiplas camadas e redes recorrentes (Haykin, 1999).

As redes neurais de uma camada são a maneira mais simples de se estruturar uma rede em camadas. Elas consistem de uma camada de entrada que tem suas saídas conectadas a uma camada de neurônios de saída. A camada de entrada simplesmente propaga os sinais de entrada para a camada com os neurônios de saída. As redes neurais de múltiplas camadas diferenciam-se das redes neurais de uma camada pelo acréscimo de uma ou mais camadas intermediárias de neurônios entre a camada de sinais de entrada e a camada de neurônios de saída. Os sinais são propagados de uma camada à outra, até atingirem a camada de saída, produzindo a saída da rede neural. Um tipo de rede recorrente caracteriza-se pela propagação dos sinais de saída de uma camada para as entradas dos neurônios da mesma camada, podendo um neurônio receber ou não seu próprio sinal de saída como entrada. A Figura 2.8 apresenta as arquiteturas de redes neurais de uma e múltiplas camadas. As próximas seções apresentam as redes neurais mais utilizadas em problemas de classificação, chamadas *Single-Layer Perceptron* (SLP) e *Multi-Layer Perceptron* (MLP).

Figura 2.8: Arquiteturas de redes neurais: (a) uma camada; (b) múltiplas camadas.

2.2.4 Single-Layer Perceptron

O *perceptron* (Rosenblatt, 1958, 1962) foi o primeiro modelo de neurônio artificial que utiliza aprendizado supervisionado, ou seja, aprende com o auxílio de um professor ou supervisor que possui conhecimento sobre o problema a ser solucionado. Assim, o *perceptron* pode ser utilizado para a tarefa de classificação por meio do conhecimento sendo apresentado ao neurônio como um conjunto de exemplos de treinamento contendo suas saídas (classes) desejadas.

O *perceptron* utiliza o modelo de neurônio proposto por McCulloch e Pitts. O neurônio calcula uma combinação linear das entradas aplicadas, incorporando um bias externo. A soma resultante é então aplicada à função de ativação. De acordo com a saída produzida pelo neurônio, seus pesos sinápticos são atualizados. A adaptação dos pesos sinápticos do *perceptron* a uma matriz de exemplos de treinamento $\mathbf{X}$ acontece da seguinte maneira (Haykin, 1999):

1. **Inicialização:** inicializar os pesos sinápticos ($\mathbf{w}$) aleatoriamente com valores pequenos;
2. **Apresentação dos exemplos de treinamento:** para cada exemplo de treinamento $\mathbf{x}_i$, executar os passos 3 e 4;
3. **Cálculo da resposta do neurônio:** a saída do neurônio para um exemplo de entrada $\mathbf{x}_i$ é calculada utilizando a Equação (2.3), em que $f(\cdot)$ é a função de ativação:

$$y(t) = f(\mathbf{w} \times \mathbf{x}_i) \quad (2.3)$$

4. **Atualização do vetor de pesos:** os pesos sinápticos são atualizados de acordo com a Equação (2.4):

$$\mathbf{w}(t + 1) = \mathbf{w}(t) + \eta[d_i - y_i]\mathbf{x}_i \quad (2.4)$$

5. **Continuação:** Repetir passos 2, 3 e 4 até que um critério de parada seja satisfeito.

No processo de aprendizagem do *perceptron*, η representa a taxa de aprendizado do *perceptron* e $d_i - y_i$ representa o erro obtido pela diferença entre a saída do neurônio (y_i) e a saída desejada (d_i) do exemplo $\mathbf{x}_i$. Esses mesmos passos de aprendizado podem ser utilizados se for considerada uma camada formada por vários *perceptrons*, de forma a obter múltiplas saídas. Basta apenas utilizar uma matriz de pesos $\mathbf{W}$ e uma matriz de saídas $\mathbf{D}$, em que cada coluna de $\mathbf{D}$ corresponde à saída desejada de um exemplo de treinamento.

Apesar da eficiência do *perceptron* formado por uma camada, o trabalho de Minsky e Papert (1969) demonstrou que esse modelo é incapaz de solucionar problemas cujas classes não sejam linearmente separáveis, ou seja, para o *perceptron* funcionar corretamente, os exemplos a serem classificados devem estar separados suficientemente uns dos outros de maneira a garantir que a superfície de separação entre eles seja um hiperplano (Haykin, 1999). Como exemplos de problemas linearmente e não linearmente separáveis, podem-se citar os operadores lógicos *AND* e *XOR*, respectivamente, ilustrados na Figura 2.9. Como pode ser observado na Figura 2.9(b), não existe um hiperplano, no caso uma reta, que separe os exemplos do problema.

Figura 2.9: Exemplos de problemas linearmente e não linearmente separáveis: (a) função lógica *AND*; (b) função lógica *XOR*. Adaptado de (de Castro, 2006).

2.2.5 Multi-Layer Perceptron e Back-propagation

O trabalho de Minsky e Papert (1969) causou um grande impacto na comunidade científica, fazendo com que o interesse no estudo de RNA diminuísse muito durante a década de 70. Entretanto, após a publicação dos dois volumes do livro *Parallel distributed processing* (Rumelhart e McClelland, 1986; McClelland et al., 1986) propondo a utilização do algoritmo *Back-propagation* em tarefas de Aprendizado de Máquina (AM) e demonstrando seu funcionamento, o interesse em RNA voltou a crescer na comunidade científica.

Figura 2.10: Fluxo do sinal em um neurônio j . Adaptado de (Haykin, 1999).

Considere a Figura 2.10, na qual v_j é a saída do neurônio j , dada como entrada à função de ativação associada ao neurônio, e y_j é a saída do neurônio j após a aplicação da função de ativação φ . A ideia básica do processo de aprendizado do *Back-propagation* é a repetida aplicação da regra da cadeia do Cálculo, de modo a calcular a influência de cada peso sináptico da rede no processo de aprendizagem. Essa influência é calculada com relação a uma função de erro $\mathcal{E}$ de todos os parâmetros livres da rede (pesos sinápticos e bias). Assim, a função $\mathcal{E}$ é uma medida de desempenho, e o objetivo do aprendizado do *Back-propagation* é ajustar esses parâmetros livres com o objetivo de minimizar a função $\mathcal{E}$. Os pesos da rede são ajustados proporcionalmente ao valor da derivada parcial $\partial\mathcal{E}/\partial w_{ji}$. Essa derivada parcial é dada na Equação 2.5.

$$\frac{\partial \mathcal{E}}{\partial w_{ji}} = \frac{\partial \mathcal{E}}{\partial e_j} \frac{\partial e_j}{\partial y_j} \frac{\partial y_j}{\partial v_j} \frac{\partial v_j}{\partial w_{ji}} \quad (2.5)$$

A derivada parcial representa um fator de sensibilidade, e determina a direção da busca pelo peso sináptico w_{ji} no espaço de pesos. Acompanhando o fluxo do sinal em um neurônio j (Figura 2.10), pode-se obter os valores das derivadas da Equação 2.5 (Haykin, 1999):

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial e_j} &= e_j & \frac{\partial e_j}{\partial y_j} &= -1 \\ \frac{\partial y_j}{\partial v_j} &= \varphi'_j(v_j) & \frac{\partial v_j}{\partial w_{ji}} &= y_i \end{aligned} \quad (2.6)$$

Uma vez conhecidas as derivadas parciais de cada peso sináptico, o objetivo do aprendizado é minimizar a função de erro $\mathcal{E}$ por meio da regra delta, dada pela Equação 2.7. Na Equação, η representa a taxa de aprendizado do algoritmo *Back-propagation*. A utilização do sinal de menos indica a execução do gradiente descendente no espaço de busca, ou seja, a procura por uma direção para a mudança do peso sináptico que reduza o erro $\mathcal{E}(t)$ (Haykin, 1999).

$$w_{ji}(t+1) = w_{ji}(t) - \eta \frac{\partial \mathcal{E}(t)}{\partial w_{ji}(t)} \quad (2.7)$$

Basicamente, o processo de aprendizado do algoritmo *Back-propagation* consiste de dois passos: uma fase de propagação e uma fase de retropropagação. Na fase de propagação, um exemplo de entrada é aplicado aos neurônios de entrada da rede, e os sinais produzidos por esses neurônios são propagados camada a camada até que uma saída seja produzida pela rede. Após essa fase, o erro da rede é calculado subtraindo-se a saída desejada (d) da saída obtida (o) pela rede. Esse erro é então retropropagado pela rede camada a camada, e os pesos da rede são ajustados para tornar a saída da rede próxima da saída desejada. O algoritmo *Back-propagation* pode ser descrito da seguinte maneira (Haykin, 1999):

1. **Inicialização:** inicializar os pesos sinápticos com valores pequenos provenientes de uma distribuição uniforme;
2. **Apresentação dos exemplos de treinamento:** misturar os exemplos aleatoriamente e apresentá-los à rede. Para cada exemplo do conjunto de treinamento, executar as fases de propagação e retropropagação descritas nos passos 3 e 4;
3. **Fase de propagação:** seja um exemplo de treinamento denotado por uma tupla $(\mathbf{x}(t), \mathbf{d}(t))$, com $\mathbf{x}$ sendo um vetor de atributos de um exemplo de entrada e $\mathbf{d}$ um vetor com as saídas desejadas do exemplo. Os sinais de saída de cada neurônio devem ser calculados e propagados camada a camada. A saída $v_j^{(l)}(t)$ de um neurônio j na camada l é obtida por meio da Equação (2.8):

$$v_j^{(l)}(t) = \sum_{i=0}^m w_{ji}^{(l)}(t) y_i^{(l-1)}(t) \quad (2.8)$$

Na Equação (2.8), $y_i^{(l-1)}(t)$ é a saída do neurônio i na camada anterior $l - 1$ na iteração t , e $w_{ji}^{(l)}(t)$ é o peso sináptico do neurônio j na camada l conectado ao neurônio i na camada $l - 1$. Para $i = 0$, têm-se $y_0^{(l-1)}(t) = +1$ e $w_{j0}^{(l)}(t) = b_j^{(l)}(t)$, sendo $b_j^{(l)}(t)$ o bias aplicado ao neurônio j na camada l . Com o uso de uma função de ativação φ , a saída do neurônio j na camada l é dada pela Equação (2.9):

$$y_j^{(l)} = \varphi_j(v_j(t)) \quad (2.9)$$

Se o neurônio j estiver localizado na primeira camada intermediária, ou seja, $l = 1$, então $y_j^{(0)}(t) = x_j(t)$, em que $x_j(t)$ é o j -ésimo elemento do vetor de entrada $\mathbf{x}(t)$. Se o neurônio j estiver na camada de saída, ou seja, $l = L$, em que L é o número de camadas da rede, então $y_j^{(L)} = o_j(t)$, e o erro do neurônio (e_j) deve ser calculado de acordo com a Equação (2.10), em que $d_j(t)$ é o j -ésimo elemento do vetor de saídas desejadas $\mathbf{d}(t)$, e $o_j(t)$ é o j -ésimo elemento do vetor de saídas verdadeiras $\mathbf{o}(t)$.

$$e_j(t) = d_j(t) - o_j(t) \quad (2.10)$$

O valor instantâneo do erro $e_j(t)$ de um neurônio j no tempo t é definido por $1/2e_j^2$. O valor instantâneo $\mathcal{E}_s(t)$ do erro total obtido pela rede considerando todos os neurônios da camada de saída é obtido somando os valores $1/2e_j^2$ dos erros de todos os neurônios da camada. Esses são os únicos neurônios considerados visíveis, pois seus erros podem ser calculados diretamente. Esse cálculo é mostrado na Equação 2.11, na qual C é o conjunto de todos os neurônios da camada de saída.

$$\mathcal{E}_s(t) = \frac{1}{2} \sum_{j \in C} e_j^2(t) \quad (2.11)$$

O erro total da rede considerando todos os exemplos, ou erro quadrático médio $\mathcal{E}_m$, é obtido por meio da soma do erro $\mathcal{E}_s(t)$ obtido em todos os exemplos de treinamento, e então normalizando essa soma com relação ao número de exemplos N . Esse cálculo é apresentado na Equação 2.12.

$$\mathcal{E}_m = \frac{1}{N} \sum_{n=1}^N \mathcal{E}_s(t) \quad (2.12)$$

4. **Fase de retropropagação:** Nessa fase são calculados os gradientes locais da rede (δ), definidos da seguinte maneira:

$$\delta_j^{(l)}(t) = \begin{cases} e_j^{(L)}(t) \varphi_j'(v_j^{(L)}(t)) & (1) \\ \varphi_j'(v_j^{(l)}(t)) \sum_k \delta_k^{(l+1)}(t) w_{kj}^{(l+1)}(t) & (2) \end{cases}$$

(1) se o neurônio j estiver na camada de saída L .

(2) se o neurônio j estiver na camada intermediária l .

No cálculo dos gradientes locais, $\varphi_j'(\cdot)$ denota a derivada da função de ativação com relação ao seu argumento. Os pesos sinápticos da rede na camada l são então ajustados de acordo com a regra apresentada na Equação (2.13), em que η é a taxa de aprendizado e α é a constante *momentum*.

$$w_{ji}^{(l)}(t+1) = w_{ji}^{(l)}(t) + \alpha[w_{ji}^{(l)}(t-1)] + \eta \delta_j^{(l)}(t) y_i^{(l-1)}(t) \quad (2.13)$$

5. **Iteração:** as fases de propagação e retropropagação (passos 3 e 4) devem ser repetidas, até que um critério de parada seja satisfeito.

Taxa de Aprendizado e Constante Momentum

A taxa de aprendizado η utilizada tem influência direta na trajetória percorrida pelo gradiente descendente no espaço de busca (espaço de pesos). Quanto menor o valor de η , menores serão as mudanças dos pesos sinápticos de uma iteração para outra do algoritmo *Back-propagation*, e mais suave será a trajetória percorrida. Entretanto, muitos passos do algoritmo serão necessários até que uma solução aceitável seja encontrada. Se uma taxa de aprendizado muito alta for utilizada, o processo de aprendizado é acelerado. No entanto, essa aceleração é resultado de mudanças grandes nos valores dos pesos sinápticos, o que faz com que a rede se torne instável.

Com o objetivo de aumentar a taxa de aprendizado e ainda assim prevenir instabilidade na rede, é utilizada uma constante *momentum* (Rumelhart e McClelland, 1986). A constante *momentum* leva em consideração a mudança do peso sináptico no passo anterior do algoritmo para influenciar a mudança do peso no passo atual. O objetivo é acelerar a convergência do algoritmo em direções de declive constante no espaço de busca (derivada parcial com mesmo sinal durante iterações consecutivas), e ter um efeito estabilizador em regiões nas quais o sinal da derivada parcial se mantém oscilando (derivada parcial com sinal oposto durante iterações consecutivas) (Haykin, 1999).

Função de Ativação

O cálculo da derivada parcial $\partial \mathcal{E}(t) / \partial w_{ji}(t)$ requer o cálculo do gradiente δ de cada neurônio, que por sua vez requer o conhecimento da derivada da função de ativação $\varphi_j'(\cdot)$ associada ao neurônio. Para que essa derivada exista, a função de ativação deve ser contínua e diferenciável (Haykin, 1999). Uma função não linear, contínua e diferenciável, é a sigmoideal. Duas formas muito utilizadas são descritas aqui: a logística e a tangente hiperbólica.

- **Função Logística:** nessa função, a saída do neurônio j é dada pela Equação 2.14. De acordo com a equação, o sinal de saída estará sempre na faixa $0 \leq y_j \leq 1$.

$$\varphi_j(v_j(t)) = \frac{1}{1 + \exp(-av_j(t))}, \quad a > 0 \text{ e } -\infty < v_j(t) < \infty \quad (2.14)$$

A derivada de $\varphi_j(v_j(t))$ com relação a $v_j(t)$ é dada pela Equação 2.15.

$$\varphi'_j(v_j(t)) = ay_j(t)[1 - y_j(t)] \quad (2.15)$$

Quando o neurônio j for um neurônio de saída, tem-se que $y_j(t) = o_j(t)$, e o gradiente do neurônio j pode então ser calculado pela Equação 2.16.

$$\delta_j(t) = e_j(t)\varphi'_j(v_j(t)) = a[d_j(t) - o_j(t)]o_j(t)[1 - o_j(t)] \quad (2.16)$$

Se o neurônio j estiver localizado em uma camada escondida, o gradiente é dado pela Equação 2.17.

$$\delta_j(t) = \varphi'_j(v_j(t)) \sum_k \delta_k(t)w_{kj}(t) = ay_j(t)[1 - y_j(t)] \sum_k \delta_k(t)w_{kj}(t) \quad (2.17)$$

- **Função Tangente Hiperbólica:** essa função tem a forma apresentada na Equação 2.18, na qual a e b são constantes.

$$\varphi_j(v_j(t)) = a \tanh(bv_j(t)), \quad (a, b) > 0 \quad (2.18)$$

A derivada da função com relação a $v_j(t)$ é dada pela Equação 2.19.

$$\varphi'_j(v_j(t)) = \frac{b}{a}[a - y_j(t)][a + y_j(t)] \quad (2.19)$$

Se o neurônio j estiver localizado na camada de saída da rede, seu gradiente é calculado utilizando a Equação 2.20.

$$\delta_j(t) = e_j(t)\varphi'_j(v_j(t)) = \frac{b}{a}[d_j(t) - o_j(t)][a - o_j(t)][a + o_j(t)] \quad (2.20)$$

Se o neurônio j estiver localizado em uma camada escondida, o gradiente é dado pela Equação 2.21.

$$\delta_j(t) = \varphi'_j(v_j(t)) \sum_k \delta_k(t)w_{kj}(t) = \frac{b}{a}[a - y_j(t)][a + y_j(t)] \sum_k \delta_k(t)w_{kj}(t) \quad (2.21)$$

Modo de Treinamento

O treinamento de uma rede neural **MLP** se dá por meio da apresentação de exemplos de treinamento, sendo que para cada exemplo, são executadas as fases de propagação e retropropagação. A apresentação de todos os exemplos de treinamento à rede durante a fase de treinamento é chamada de época ou ciclo. O processo de aprendizado dura então um número de ciclos pré-definidos, ou até que os pesos da rede se estabilizem e o erro da mesma diminua até um valor mínimo. Além disso, a ordem de apresentação dos exemplos à rede deve ser aleatória de época para época. Isso faz com que a busca no espaço de pesos torne-se estocástica, melhorando a estabilidade da rede (Haykin, 1999). A apresentação dos exemplos para o treinamento de uma rede neural utilizando *Back-propagation* pode ser feita de duas maneiras: de modo sequencial e de modo *batch*.

No modo de treinamento sequencial, o ajuste dos pesos sinápticos da rede é feito após a apresentação de cada exemplo. Assim, quando o primeiro exemplo de um ciclo é apresentado à rede, são executadas as fases de propagação e retropropagação para o exemplo, e os pesos são ajustados de acordo com a saída da rede para esse exemplo. O mesmo procedimento é executado para todos os exemplos do conjunto de dados, até o último exemplo do ciclo ser apresentado à rede. Esse processo de apresentação de exemplos continua por vários ciclos, enquanto durar o treinamento da rede. No modo de treinamento *batch*, diferentemente, os pesos sinápticos são ajustados apenas após a apresentação de todos os exemplos de treinamento à rede (um ciclo).

O modo de treinamento sequencial é preferível ao modo de treinamento *batch* porque a atualização dos pesos a cada apresentação de um exemplo faz com que a busca no espaço de pesos torne-se naturalmente estocástica, dado que os exemplos são apresentados aleatoriamente à rede. Isso faz com que a busca do algoritmo *Back-propagation* fique menos suscetível a ficar presa em mínimos locais (Haykin, 1999).

2.2.6 Resilient Back-propagation

Como visto na seção anterior, a escolha da taxa de aprendizado η no algoritmo *Back-propagation* tem importante efeito no aprendizado da rede, pois um valor baixo leva à demora no aprendizado enquanto um valor alto leva à oscilação. Para evitar esse problema, um termo denominado constante de *momentum* é utilizado, de maneira a estabilizar o processo de aprendizagem e acelerar o mesmo em determinadas regiões do espaço de busca. Entretanto, a escolha do valor da constante de *momentum* é tão dependente da aplicação quanto à escolha do valor da taxa de aprendizado, o que faz com que muitas vezes grandes melhoras no algoritmo não sejam obtidas (Riedmiller e Braun, 1993).

O tamanho da atualização de um peso sináptico Δw_{ji} (tamanho do passo dado no espaço de busca) não é apenas dependente dos valores de taxa de aprendizado e constante *momentum*, mas também é dependente do valor da derivada parcial $\partial \mathcal{E} / \partial w_{ji}$. Assim, o efeito de uma escolha cuidadosa da taxa de aprendizado e constante *momentum* pode ser afetado pelo comportamento imprevisível da derivada parcial (Riedmiller e Braun, 1993). Assim, com o objetivo de evitar

esse problema, **Riedmiller e Braun (1993)** propuseram o algoritmo *Resilient Back-propagation* (Rprop), que modifica o tamanho da atualização de um peso sináptico Δw_{ji} diretamente, ou seja, sem considerar o valor da derivada parcial.

O algoritmo Rprop realiza uma adaptação nos pesos sinápticos baseado na informação do gradiente local dos neurônios. Ao invés de utilizar o valor da derivada parcial, cada peso tem um valor de adaptação individual Δ_{ji} . Esse valor é modificado durante o aprendizado, baseado no sinal da derivada parcial correspondente ao peso w_{ji} . O valor de Δ_{ji} é atualizado de acordo com as regras na Equação 2.22, na qual $0 < \eta^- < 1 < \eta^+$.

$$\Delta_{ji}(t) = \begin{cases} \eta^+ \times \Delta_{ji}(t-1) & , \quad \text{se } \frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}}(t) > 0 \\ \eta^- \times \Delta_{ji}(t-1) & , \quad \text{se } \frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}}(t) < 0 \\ \Delta_{ji}(t-1) & , \quad \text{caso contrário} \end{cases} \quad (2.22)$$

De acordo com a Equação 2.22, o valor de atualização Δ_{ji} é decrementado por uma fator η^- toda vez que a derivada parcial do peso w_{ji} correspondente muda de sinal. Essa mudança no sinal da derivada indica que a última atualização do peso foi muito grande e a busca “pulou” uma região de mínimo local. Se o sinal da derivada se mantém em duas iterações consecutivas, o valor Δ_{ji} é levemente aumentado por uma fator η^+ com o objetivo de acelerar a convergência em regiões planas do espaço de busca (**Riedmiller e Braun, 1993**).

Uma vez obtidos os valores de atualização de cada peso sináptico, o procedimento de atualização é executado também de acordo com o sinal da derivada parcial de cada peso. Se o sinal da derivada $\partial \mathcal{E} / \partial w_{ji}$ não mudar e for positivo, o que indica um aumento do erro, o peso w_{ji} é decrementado pelo valor Δ_{ji} . Do contrário, se o sinal da derivada não mudar mas for negativo, o valor de atualização Δ_{ji} é adicionado ao peso w_{ji} (**Riedmiller e Braun, 1993**). Essas regras são apresentadas na Equação 2.23.

$$\Delta w_{ji}(t) = \begin{cases} -\Delta_{ji}(t) & , \quad \text{se } \frac{\partial \mathcal{E}}{\partial w_{ji}}(t) > 0 \\ +\Delta_{ji}(t) & , \quad \text{se } \frac{\partial \mathcal{E}}{\partial w_{ji}}(t) < 0 \\ 0 & , \quad \text{caso contrário} \end{cases} \quad (2.23)$$

A mudança do sinal da derivada parcial de uma iteração para outra significa que a atualização anterior do peso foi muito grande, fazendo com que o mínimo da função fosse perdido na busca (**Riedmiller e Braun, 1993**). Assim, a mudança feita anteriormente no peso w_{ji} deve ser revertida. Isso é feito com a utilização da Equação 2.24.

$$\Delta w_{ji}(t) = -\Delta w_{ji}(t-1) \quad , \quad \text{se } \frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}}(t) < 0 \quad (2.24)$$

A reversão da mudança realizada anteriormente no peso w_{ji} é chamada de *backtracking* no algoritmo Rprop. Essa mudança faz com que o sinal da derivada parcial mude novamente na próxima iteração do algoritmo, punindo novamente o valor da atualização. Para evitar isso, a derivada parcial $\partial \mathcal{E} / \partial w_{ji}(t)$ recebe o valor zero para que não haja adaptação no valor de $\Delta_{ji}(t)$ novamente (**Riedmiller e Braun, 1993**). Isso pode ser visto como um truque de implementação

do algoritmo, já que faz com que a cláusula `caso contrário` na Equação 2.22 se torne ativa (Igel e Husken, 2000). Os novos pesos são então atualizados de acordo com a Equação 2.25.

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t) \quad (2.25)$$

Diferentemente do algoritmo *Back-propagation*, a atualização dos pesos no algoritmo Rprop é feita apenas depois da apresentação de todos os exemplos de treinamento à rede, ou seja, após um ciclo completo (aprendizado modo *batch*). O procedimento de atualização de pesos do Rprop é apresentado no Algoritmo 2 (Riedmiller e Braun, 1993).

Algoritmo 2: Atualização de Pesos do Algoritmo Rprop.

procedimento Rprop()

para todos os pesos e biases **faça**

se $\frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}(t)} > 0$ **então**
 $\Delta_{ji}(t) = \text{minimo}(\Delta_{ji}(t-1) \times \eta^+, \Delta_{max})$
 $\Delta w_{ji}(t) = -\text{sinal}(\frac{\partial \mathcal{E}}{\partial w_{ji}(t)} \times \Delta_{ji}(t))$
 $w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t)$

senão se $\frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}(t)} < 0$ **então**
 $\Delta_{ji}(t) = \text{maximo}(\Delta_{ji}(t-1) \times \eta^-, \Delta_{min})$
 $w_{ji}(t+1) = w_{ji}(t) - \Delta w_{ji}(t)$
 $\frac{\partial \mathcal{E}}{\partial w_{ji}(t)} = 0$

senão se $\frac{\partial \mathcal{E}}{\partial w_{ji}}(t-1) \times \frac{\partial \mathcal{E}}{\partial w_{ji}(t)} = 0$ **então**
 $\Delta w_{ji}(t) = -\text{sinal}(\frac{\partial \mathcal{E}}{\partial w_{ji}(t)} \times \Delta_{ji}(t))$
 $w_{ji}(t+1) = w_{ji}(t) - \Delta w_{ji}(t)$

2.3 Considerações Finais

Neste capítulo, foram abordados os conceitos de Redes Neurais Artificiais, que são algoritmos inspirados no funcionamento do cérebro humano. Foram apresentados alguns conceitos do sistema nervoso humano e também conceitos do neurônio artificial utilizado nas redes neurais artificiais. Também foram introduzidos os conceitos de *Single e Multi-Layer Perceptrons* e dois algoritmos utilizados no treinamento de redes neurais *Multi-Layer Perceptrons*.

Além das redes neurais, foram apresentados os conceitos de Algoritmos Genéticos, que têm como objetivo solucionar problemas utilizando como inspiração a teoria da evolução das espécies. Foi apresentado o Algoritmo Evolutivo tradicional, e também os principais operadores utilizados no processo evolutivo dos Algoritmos Genéticos. O próximo capítulo apresentará os conceitos fundamentais de classificação hierárquica multirrótulo, problema investigado nesta pesquisa de Doutorado.

Classificação Hierárquica e Multirrótulo

Este capítulo apresenta os conceitos de classificação hierárquica e classificação multirrótulo. Inicialmente, nas Seções 3.1 e 3.2, são introduzidos os conceitos de classificação hierárquica simples-rótulo e classificação multirrótulo não hierárquica. Esses dois tipos de problemas podem ser combinados, dando origem a problemas de classificação mais complexos, chamados de problemas de classificação hierárquica multirrótulo. Esses problemas são formalmente definidos na Seção 3.3; os trabalhos mais recentes da literatura que tratam problemas hierárquicos multirrótulo são revisados na Seção 3.4, separados em métodos que utilizam e que não utilizam abordagens bioinspiradas; na Seção 3.5, é feita uma revisão das principais medidas de avaliação de classificadores hierárquicos multirrótulo encontradas na literatura; por fim, a Seção 3.6 apresenta as considerações finais deste capítulo.

3.1 Classificação Hierárquica Simples-Rótulo

Em grande parte dos problemas de classificação descritos na literatura, um classificador atribui uma única classe a cada exemplo $\mathbf{x}_i$, e essas classes formam uma estrutura não hierárquica, na qual não são considerados relacionamentos de superclasses e subclasses. Entretanto, em muitos problemas de classificação reais, uma ou mais classes podem ser divididas em subclasses ou agrupadas em superclasses. Nesses casos, as classes são estruturadas hierarquicamente, geralmente em uma estrutura do tipo árvore ou do tipo grafo acíclico direcionado (DAG). Esses tipos de problemas são conhecidos na literatura de AM como problemas de classificação hierárquica. Nesses problemas, novos exemplos são atribuídos a classes associadas a nós pertencentes a uma estrutura hierárquica. Quando cada novo exemplo deve ser atribuído a um nó folha de uma hierarquia de classes, a tarefa de classificação hierárquica é denominada classificação obrigatória em nós folhas. Quando a classe mais específica atribuída a um exemplo pode ser um nó interno (não folha) da hierarquia de classes, a tarefa de classificação é denominada

classificação não obrigatória em nós folhas (Freitas e Carvalho, 2007). Nos trabalhos de Sun e Lim (2001) e Sun et al. (2003), quando a tarefa de classificação em uma estrutura em árvore é não obrigatória em nós folhas, a estrutura hierárquica é chamada de árvore virtual. Do mesmo modo, quando a mesma tarefa de classificação é executada em uma estrutura em DAG, essa estrutura é chamada DAG virtual.

Problemas hierárquicos podem ser tratados utilizando duas abordagens, chamadas local (ou *top-down*) e global (ou *one-shot*). A abordagem global também pode ser encontrada em alguns trabalhos com a denominação de *big-bang*. Na abordagem local, algoritmos de classificação convencionais são treinados para produzir uma hierarquia de classificadores, que são então aplicados utilizando uma estratégia *top-down* para a classificação de novos exemplos. Inicialmente, a classe mais genérica de um novo exemplo é predita. Essa classe, localizada no primeiro nível da hierarquia, é então utilizada para reduzir o conjunto de possíveis classes do próximo nível, que são suas subclasses. Dessa forma, durante a classificação de um exemplo, quando o mesmo é atribuído a uma classe que não é um nó folha, ele é posteriormente classificado em uma subclasse dessa classe. Uma desvantagem dessa abordagem é que, conforme a hierarquia é percorrida em direção aos nós folhas, erros de classificação cometidos são propagados para os níveis mais profundos, a não ser que algum procedimento seja adotado para evitar esse problema. Por outro lado, a abordagem possui o aspecto positivo da utilização de qualquer algoritmo de classificação convencional nos nós da hierarquia de classes.

O trabalho de Sun et al. (2004) apontou outro problema da abordagem local, chamado problema do bloqueio (*blocking problem*). Esse problema acontece quando, em um determinado momento da classificação *top-down*, um exemplo não é classificado em nenhum nó interno da hierarquia, o que faz com que ele deixe de ser classificado em classes mais específicas. Esse é um dos motivos que podem fazer com que os classificadores baseados na abordagem local obtenham baixas taxas de acerto nas classes mais profundas da hierarquia.

De acordo com Silla e Freitas (2010), o aspecto mais importante da abordagem local é que a hierarquia de classes é levada em consideração por meio da utilização de informações locais, e que essas informações podem ser utilizadas de diferentes maneiras. Assim, parecem existir três principais diferentes grupos de classificadores baseados na abordagem local, caracterizados pela maneira como as informações locais são utilizadas e pela maneira como os classificadores são induzidos: um classificador local por nó, um classificador local por nó pai, e um classificador local por nível. No primeiro grupo, um classificador binário é treinado para cada nó da hierarquia de classes, exceto para o nó raiz. A abordagem que utiliza um classificador por nó pai treina um classificador multiclasse para cada nó interno da hierarquia, ou então utiliza algum método decomposicional com classificadores binários, como o Um-Contra-Todos ou *Support Vector Machines* (SVM) (Vapnik, 1999) binários, com a finalidade de fazer a distinção entre suas subclasses. Já o último grupo treina um classificador multiclasse para cada nível da hierarquia, sendo cada classificador responsável pelas predições das classes de seu nível correspondente.

Na abordagem global, ao contrário da abordagem local, um único classificador é induzido utilizando toda a hierarquia de classes de uma única vez. Após o treinamento do classificador,

a classificação de um novo exemplo ocorre em apenas um passo. Como métodos baseados na abordagem global precisam considerar as peculiaridades do problema de classificação, algoritmos de classificação convencionais não podem ser utilizados, a não ser que sejam adaptados para considerar toda a hierarquia de classes. Como resultado, esses métodos geralmente possuem uma maior complexidade de implementação, mas evitam o problema de propagação de erros da abordagem local. Adicionalmente, se a abordagem global for utilizada para gerar regras de classificação, o conjunto de regras globais gerado tende a ser menos complexo (possuindo menos regras) do que a coleção de todos os conjuntos de regras locais gerados pelos classificadores utilizados na abordagem local (Vens et al., 2008; Alves et al., 2010; Otero et al., 2010).

Diferentemente da abordagem local, classificadores baseados na abordagem global não conseguem utilizar informações locais durante a classificação de novos exemplos. Como as características utilizadas para classificar um exemplo em classes próximas à raiz são diferentes das características utilizadas para classificar o mesmo exemplo em classes mais profundas, geralmente um classificador baseado na abordagem global tem dificuldades na exploração de diferentes conjuntos de características em diferentes níveis hierárquicos (Sun e Lim, 2001).

A Figura 3.1 ilustra exemplos de dois problemas de classificação de artigos científicos, sendo um deles hierarquicamente estruturado como uma árvore e o outro hierarquicamente estruturado como um DAG. A principal diferença entre o problema estruturado como uma árvore (Figura 3.1(a)) e o problema estruturado como um DAG (Figura 3.1(b)) é que, na estrutura em árvore, cada nó, exceto o nó raiz, possui exatamente um nó pai, enquanto que na estrutura em DAG, cada nó pode possuir mais de um nó pai. Consequentemente, em problemas estruturados como árvores, cada classe possui uma única profundidade, porque há apenas um possível caminho entre o nó raiz e qualquer outro nó da hierarquia. Diferentemente, em problemas estruturados como DAGs, a profundidade de uma classe não é mais única, pois pode haver mais de um caminho entre o nó raiz e qualquer nó da hierarquia. Essas características devem ser consideradas durante o desenvolvimento e avaliação de classificadores hierárquicos.

Figura 3.1: Problemas de classificação de artigos científicos: (a) hierarquia estruturada como uma árvore; (b) hierarquia estruturada como um DAG.

De maneira geral, uma hierarquia estruturada como um DAG pode ser interpretada de duas maneiras. A primeira considera que se um exemplo é atribuído a uma classe c_j , ele consequentemente é atribuído a todas as classes pais da classe c_j . Outra interpretação considera que se

um exemplo é atribuído a uma classe c_j , ele deve ser atribuído a pelo menos uma subclasse da classe c_j . Intuitivamente, a primeira interpretação parece mais correta, já que respeita estritamente a hierarquia de classes. Adicionalmente, essa é a interpretação correta para a hierarquia de classes da *Gene Ontology* (GO).

Em ambas as estruturas hierárquicas em árvore ou DAG, a tarefa de predição de classes associadas aos nós mais profundos da hierarquia geralmente apresenta baixas taxas de acerto. Isso ocorre porque essas classes geralmente possuem poucos exemplos positivos, o que dificulta a tarefa de aprendizado dos classificadores. Essas classes também trazem informações mais específicas sobre os exemplos atribuídos a elas.

3.2 Classificação Multirrótulo não Hierárquica

Em problemas de classificação multirrótulo, cada exemplo $\mathbf{x}_i$ pode ser associado a duas ou mais classes simultaneamente. Um classificador multirrótulo pode ser representado por uma função $H : \mathbf{X} \rightarrow 2^C$, que mapeia um exemplo $\mathbf{x}_i \in \mathbf{X}$ (espaço de exemplos) em um subconjunto de classes $C_i \in C$ (conjunto de classes).

A Figura 3.2 ilustra uma comparação entre um problema de classificação convencional, no qual exemplos podem ser atribuídos a apenas uma classe, e um problema de classificação multirrótulo. A Figura 3.2(a) ilustra um problema de classificação em que um documento pode pertencer a apenas uma dentre duas classes (“Biologia” ou “Ciência da Computação”), mas nunca a ambas as classes simultaneamente. A Figura 3.2(b) ilustra um problema de classificação em que um documento pode ser atribuído simultaneamente às classes “Biologia” e “Ciência da Computação”. Na Figura 3.2(b), os exemplos dentro da região marcada abordam assuntos tanto de Biologia quanto de Ciência da Computação.

Figura 3.2: Exemplos de problemas de classificação: (a) problema de classificação convencional (simples-rótulo); (b) problema de classificação multirrótulo. Adaptado de (Shen et al., 2003).

Similarmente aos problemas hierárquicos, nos quais as abordagens global e local podem ser utilizadas para resolver a tarefa de classificação, duas abordagens podem ser utilizadas para resolver problemas de classificação multirrótulo, chamadas de dependente de algoritmo e independente de algoritmo (Carvalho e Freitas, 2009). A abordagem independente de algoritmo

transforma o problema multirrótulo em um conjunto de problemas simples-rótulo e, como na abordagem local, qualquer algoritmo de classificação convencional pode ser utilizado. Na abordagem dependente de algoritmo, como o próprio nome sugere, novos algoritmos são desenvolvidos especificamente para problemas multirrótulo, ou modificações são feitas em algoritmos convencionais. A abordagem global utilizada em problemas hierárquicos pode ser vista como uma abordagem dependente de algoritmo, já que são utilizados algoritmos novos ou modificados.

Conforme afirmado por [Boutell et al. \(2004\)](#), é importante notar que problemas de classificação multirrótulo são diferentes de problemas de classificação *fuzzy*. A classificação *fuzzy* é utilizada para lidar com ambiguidades entre múltiplas classes de um mesmo exemplo, e não é utilizada para se obter uma classificação multirrótulo. Geralmente, uma etapa de defuzzificação é utilizada para se obter uma classificação final. A classificação multirrótulo, por outro lado, trata problemas em que um exemplo pode possuir todas as propriedades de múltiplas classes simultaneamente, e essas classes podem ser muito distintas. Adicionalmente, a utilização de uma função de pertinência em ambos os problemas é diferente. Enquanto em classificadores *fuzzy*, para cada exemplo, a soma dos graus de pertinência de todas as classes deve ser 1, essa restrição não é aplicada em problemas multirrótulo, em que cada exemplo pode ser atribuído a mais de uma classe com 100% de pertinência, o que equivaleria a uma soma total das pertinências das classes muito maior do que 1.

Outras características importantes em conjuntos de dados multirrótulo são a cardinalidade de rótulo e densidade de rótulo. Os conjuntos de dados não são todos igualmente multirrótulo, pois, em alguns casos, o número de classes $|C|$ atribuídas a cada exemplo é pequeno se comparado com o número total de exemplos, enquanto em outros, esse número é grande. Essa característica influencia o desempenho dos diferentes métodos de classificação multirrótulo ([Tsoumakas et al., 2010](#)). Sendo D um conjunto de dados multirrótulo consistindo de $|D|$ exemplos multirrótulo $(\mathbf{x}_i, C_i)$, com $i = 1 \dots |D|$ e $C_i \subseteq C$, pode-se definir cardinalidade e densidade de acordo com as Equações 3.1 e 3.2:

- A cardinalidade de rótulo de D é dada pelo número médio de rótulos dos exemplos de D :

$$CR(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} |C_i| \quad (3.1)$$

- A densidade de rótulo de D é dada pelo número médio de rótulos dos exemplos de D dividido por $|C|$:

$$DR(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|C_i|}{|C|} \quad (3.2)$$

De acordo com ([Tsoumakas et al., 2010](#)), a cardinalidade de rótulo de um conjunto de dados é independente de seu número de classes, e quantifica o número de rótulos alternativos que caracterizam os exemplos do conjunto de dados. A densidade de rótulo, por outro lado, leva em consideração o número de classes do conjunto de dados. Assim, dois conjuntos de dados que

possuem a mesma cardinalidade de rótulo, mas têm uma grande diferença no número de classes, ou seja, possuem diferentes densidades de rótulo, podem não ter as mesmas propriedades, influenciando no comportamento dos métodos de classificação. A relação entre a densidade e a cardinalidade de rótulo é dada por: $CR(D) = |C| \times DR(D)$.

3.3 Classificação Hierárquica Multirrótulo

Em problemas de classificação hierárquica multirrótulo, as características dos problemas hierárquicos e dos problemas multirrótulo são combinadas, e um exemplo pode ser atribuído a dois ou mais caminhos em uma hierarquia de classes. Problemas de classificação hierárquica multirrótulo foram formalmente definidos por [Vens et al. \(2008\)](#) da seguinte maneira:

Dado:

- um espaço de exemplos $\mathbf{X}$;
- uma hierarquia de classes $(C, \leq_h)$, em que C é um conjunto de classes e $\leq_h$ é uma ordem parcial 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 for uma superclasse de c_2);
- um conjunto T de tuplas $(\mathbf{x}_i, C_i)$ com $\mathbf{x}_i \in \mathbf{X}$ e $C_i \subseteq C$, tal que $c \in C_i \Rightarrow \forall c' \leq_h c : c' \in C_i$;
- um critério de qualidade q que recompensa modelos com alto desempenho preditivo e baixa complexidade.

Encontrar:

- uma função $f : \mathbf{X} \rightarrow 2^C$, em que 2^C é o conjunto potência de C , tal que $c \in f(\mathbf{x}) \Rightarrow \forall c' \leq_h c : c' \in f(\mathbf{x})$ e f otimiza q .

O critério de qualidade q pode ser o desempenho preditivo médio da classificação, ou então a distância entre as classes preditas e verdadeiras na hierarquia. Também pode ser considerado que erros de classificação em classes localizadas perto da raiz são piores do que erros cometidos em classes localizadas em níveis mais profundos, e por isso devam ser mais penalizados. Além disso, a complexidade dos classificadores e o tempo de indução também podem ser levados em consideração como critérios de qualidade.

Um exemplo de problema hierárquico multirrótulo é apresentado na Figura 3.3, em que a hierarquia de classes é estruturada como uma árvore. No exemplo da figura, um artigo científico pode ser classificado simultaneamente nas classes “Biologia/Bioestatística”, “Biologia/Bioinformática”, e “Ciência da Computação/Inteligência Artificial”. Quando uma predição é feita nos nós internos da hierarquia de classes da figura, uma subárvore é gerada. No caso da classificação hierárquica simples-rótulo, essa subárvore é reduzida a um caminho.

Figura 3.3: Problema de classificação hierárquica multirrótulo estruturado como um árvore: (a) hierarquia de classes; (b) previsões gerando uma subárvore.

É importante notar a diferença entre problemas hierárquicos simples-rótulo e problemas multirrótulo não hierárquicos. Um problema hierárquico simples-rótulo pode ser visto como sendo naturalmente multirrótulo de maneira trivial, devido ao fato de um caminho em uma hierarquia de classes possuir mais de uma classe. Quando o caminho “Biologia/Bioinformática” é atribuído a um exemplo, essa predição significa que o exemplo pertence a duas classes: “Biologia” e “Bioinformática”. Entretanto, na literatura, um problema hierárquico é considerado multirrótulo somente no caso não trivial em que classes de mais de um caminho na hierarquia são atribuídas a um exemplo.

3.4 Trabalhos Relacionados

Muitos métodos têm sido propostos na literatura para tratar problemas de classificação hierárquica multirrótulo. A maioria desses métodos é aplicada em problemas de predição de funções de genes e proteínas e em problemas de classificação de textos. Esta seção apresenta os mais recentes trabalhos propostos, organizados de acordo com a taxonomia proposta por [Silla e Freitas \(2010\)](#). Nessa taxonomia, um algoritmo de classificação hierárquica é descrito por uma 4-tupla $\langle \Delta, \Xi, \Omega, \Theta \rangle$, em que:

- Δ : indica se o algoritmo é hierárquico simples-rótulo (*Single Path Prediction (SPP)*) ou hierárquico multirrótulo (*Multiple Path Prediction (MPP)*);
- Ξ : indica o tipo de predição do algoritmo, se obrigatória ou não obrigatória em nós folhas - *Mandatory Leaf-Node Prediction (MLNP)* ou *Non-Mandatory Leaf-Node Prediction (NMLNP)*;
- Ω : indica o tipo de estrutura hierárquica na qual o algoritmo pode ser aplicado - T (estrutura em árvore) ou D (estrutura em DAG);
- Θ : indica a abordagem utilizada pelo algoritmo na taxonomia proposta, se um classificador local por nó, por nível, por nó pai ou global - *Local Classifier per Node (LCN)*, *Local Classifier per Level (LCL)*, *Local Classifier per Parent Node (LCPN)* ou *Global Classifier (GC)*.

As próximas duas seções apresentam os trabalhos revisados. Como os dois métodos propostos nesta pesquisa de Doutorado são bioinspirados, os trabalhos encontrados na literatura foram divididos em trabalhos bioinspirados e não bioinspirados. Primeiramente são revisados os trabalhos não bioinspirados, que são a maioria dos trabalhos encontrados. Os trabalhos que são bioinspirados são revisados posteriormente. A ordem de apresentação dos trabalhos segue as abordagens em que os mesmos foram baseados. Primeiro são apresentados os trabalhos que propõem métodos baseados na abordagem local, e em seguida os trabalhos que propõem métodos baseados na abordagem global.

3.4.1 Abordagens que não são Bioinspiradas

No trabalho de [Dumais e Chen \(2000\)](#), classificadores baseados em [SVMs \(Vapnik, 1999\)](#) foram treinados para cada nó da hierarquia de classes (abordagem [LCN](#)), com o objetivo de classificar documentos da Web. A hierarquia utilizada possuía apenas dois níveis, e para cada exemplo a ser classificado, os classificadores [SVM](#) individuais tinham como saída a probabilidade do exemplo pertencer a cada uma das 13 classes do primeiro nível, e a cada uma das 150 classes do segundo nível. Cada classificador foi treinado para distinguir sua classe correspondente de todas as outras classes do mesmo nível. Foram utilizadas duas maneiras de se combinar as probabilidades do primeiro e do segundo nível. Na primeira maneira, é ajustado um valor de limiar para o primeiro nível, e somente os exemplos que foram classificados no primeiro nível, passam para o segundo nível. Em um segundo caso, exemplos podem ser classificados em classes do segundo nível, mesmo sem terem sido classificados em classes do primeiro nível.

Nos trabalhos de [Sun e Lim \(2001\)](#) e [Sun et al. \(2003\)](#), diferentes medidas de avaliação de classificadores hierárquicos multirrótulo foram propostas. Essas medidas são baseadas na distância entre as classes preditas e reais na hierarquia de classes, e também nos relacionamentos semânticos (similaridade) entre as classes da hierarquia. As diferentes medidas foram utilizadas em métodos de classificação baseados na abordagem [LCN](#), utilizando classificadores [SVM](#) e Naive Bayes binários para a tarefa de classificação de documentos.

No trabalho de [Nguyen et al. \(2005\)](#), a abordagem [LCN](#) utilizada em [\(Sun e Lim, 2001\)](#) foi adaptada para a utilização em hierarquias estruturadas como [DAGs](#). Duas adaptações foram propostas. A primeira consistiu na transformação do [DAG](#) em uma estrutura em árvore equivalente. Essa transformação é feita por meio de uma busca em profundidade no grafo, sendo que cada vez que um nó é visitado, uma cópia do mesmo é indexada de acordo com o número de visitas e é então adicionada à árvore sendo construída. Após essa transformação, um classificador [SVM](#) é associado a cada nó da árvore criada. A segunda adaptação consistiu da associação de mais de um classificador a cada nó da hierarquia, de acordo com o número de pais do nó. Assim, um nó com n pais teria associado a ele n classificadores [SVM](#). Experimentos em conjuntos de dados de documentos apresentaram bons resultados.

Em [\(Sun et al., 2004\)](#), diferentes técnicas para tentar contornar o problema do bloqueio presente na abordagem local foram propostas: *Threshold Reduction Method* (TRM), *Restricted Voting Method* (RVM), e *Extended Multiplicative Method* (EMM). A técnica TRM consiste em

diminuir os valores de limiares associados a cada classificador conforme o nível da hierarquia torna-se mais profundo. Assim, é permitido que mais exemplos sejam passados para os classificadores associados aos níveis mais profundos. Na técnica RVM, um classificador tem a chance de classificar um exemplo antes que o mesmo seja classificado por outro classificador associado a sua superclasse. Assim, é criado um conjunto de classificadores secundários que consideram uma classe e suas classes avós. A técnica EMM classifica um exemplo em um classe c_j se o produto das saídas do classificador associado a c_j e do classificador associado à superclasse de c_j for maior que um dado limiar. Ou seja, os limiares são considerados a cada dois níveis. É importante ressaltar que as técnicas propostas por Sun et al. (2004) são utilizadas em métodos que têm como saída um número real, representando a probabilidade de um exemplo pertencer a uma dada classe em um dado nível. Experimentos utilizando a abordagem LCN e classificadores SVM mostraram que as técnicas propostas podem melhorar o desempenho da abordagem local. De acordo com os experimentos, os melhores resultados foram obtidos com a utilização da técnica *Restricted Voting*.

Em (Barutcuoglu et al., 2006) foi proposto um método baseado na abordagem LCN, que utiliza uma hierarquia de classificadores SVM para a predição de funções de genes. O conjunto de genes utilizado pertence à hierarquia de processos biológicos da *Gene Ontology* (GO) (Ashburner et al., 2000). Classificadores são treinados para cada classe separadamente, e as predições são então combinadas utilizando um modelo de Redes Bayesianas (Friedman et al., 1997), com o objetivo de encontrar o mais provável conjunto consistente de predições. O mesmo método utilizando redes Bayesianas foi utilizado no trabalho de Barutcuoglu e DeCoro (2006), utilizando o algoritmo *k-Nearest Neighbors* (KNN) (Aha et al., 1991) como classificador base. O método foi aplicado à tarefa de classificação de imagens, apresentando bons resultados. Ainda utilizando o mesmo modelo de combinação de classificadores, o trabalho de DeCoro et al. (2007) utilizou classificadores SVM aplicados à tarefa de classificação de músicas.

O trabalho de Guan et al. (2008) estendeu o método proposto por Barutcuoglu et al. (2006) utilizando uma combinação de vários conjuntos de dados anotados segundo a GO. O método proposto é um *framework* de combinação baseado em três diferentes métodos: (i) o treinamento de um único classificador SVM para cada termo da GO pertencente à combinação dos dados; (ii) combinação dos classificadores SVM por meio da utilização de Redes Bayesianas para corrigir suas predições baseado nos relacionamentos hierárquicos da GO; e (iii) construção de um classificador Naive Bayes (Langley et al., 1992) para cada termo da GO para integrar os resultados dos classificadores SVM independentes. A predição final foi formada pela seleção do método que obteve o melhor desempenho em cada termo da GO.

Ainda baseado na abordagem LCN, um classificador bayesiano (*Bayes Optimal Classifier*) (Bayes, 1763) chamado *HBayes* foi proposto por Cesa-Bianchi et al. (2006), e aplicado a dois conjuntos de documentos: *Reuters Corpus Volume 1* (RCV1) (Lewis et al., 2004) e uma subárvore do corpus OHSUMED de resumos médicos (Hersh et al., 1994). Nesse método, os relacionamentos entre as classes da hierarquia são interpretados como uma floresta. As árvores dessa floresta representam a taxonomia de classes. O método consiste em treinar um classifica-

dor para cada nó da hierarquia. Com isso, são calculadas estimativas $\hat{p}_j(\mathbf{x}_i)$ de probabilidades para as pertinências das classes a todos os exemplos da hierarquia. O método é iniciado colocando todos os nós da taxonomia em um conjunto. Esses nós são então removidos um a um, e cada vez que um exemplo é atribuído a uma classe c_j , ele é também atribuído às classes pertencentes ao caminho da raiz de sua árvore até a classe c_j . Um nó só pode ser removido se ele for uma folha, ou se todos os nós da sua subárvore já tiverem sido removidos.

O trabalho de [Cesa-Bianchi e Valentini \(2010\)](#) estendeu o método de *ensemble* descrito em [Cesa-Bianchi et al. \(2006\)](#), apresentando uma variante do algoritmo que, segundo os autores, é mais adequada para conjuntos de dados esparsos. Na variante proposta, *HBayes Cost Sensitive* (*HBayes-CS*), foi introduzido um parâmetro α para balancear os custos de Falsos Positivos (FP) e Falsos Negativos (FN). Experimentos utilizando conjuntos de dados biológicos mostraram que houve melhora no desempenho preditivo do método.

Um método de classificação baseado em busca estocástica foi proposto no trabalho de [Jin et al. \(2008\)](#). O método foi chamado de *Random GO Walk* (RGOW), e procura pelo melhor conjunto de classes para um dado documento baseado em um algoritmo chamado *Metropolis-Hastings* ([Hastings, 1970](#)), utilizando o procedimento *simulated annealing* ([Kirkpatrick et al., 1983](#)). Segundo os autores, o objetivo do método é explorar a busca estocástica para aliviar o problema de máximos locais presente na busca feita pelo método *top-down*. Adicionalmente, o método tem como saída uma distribuição de probabilidades sobre as classes folhas da hierarquia, refletindo uma probabilidade *a posteriori* das múltiplas classes. O método apresentou bons resultados na comparação com outros dois métodos baseados na abordagem LCN utilizando os algoritmos SVM e Naive Bayes como classificadores base.

No trabalho de [Bennett e Nguyen \(2009\)](#) foi proposto um método chamado *Experts Refinements* ainda baseado na abordagem LCN, que consiste em fazer um refinamento da fase de treinamento dos classificadores base. Esse refinamento é obtido por meio da utilização da técnica de validação cruzada para a obtenção de uma melhor estimativa das probabilidades de pertinência associadas às classes preditas. O processo de refinamento é combinado com a abordagem de treinamento *bottom-up*, treinando primeiramente classificadores SVM associados às folhas da hierarquia e passando as informações para os classificadores associados às classes pais dessas classes. O método proposto foi utilizado em um conjunto de dados de documentos da Web, e obteve resultados melhores do que quando os classificadores SVM foram utilizados sem o refinamento.

Outro método baseado na abordagem LCN foi proposto no trabalho de [Mayne e Perry \(2009\)](#), e utilizado para a tarefa de classificação de textos. Foi utilizada uma hierarquia de classificadores Naive Bayes, e cada classificador tem como saída a probabilidade do exemplo pertencer a sua classe associada. Essa probabilidade é então transmitida aos classificadores associados às classes filhas da classe em questão, e adicionada ao vetor de atributos do exemplo sendo classificado. As probabilidades de pertinência de cada classe podem ser concatenadas aos vetores de atributos dos exemplos em cada nível hierárquico.

Em ([Punera e Rajan, 2009](#)), classificadores SVM foram utilizados em um método que utiliza

a abordagem LCN. Cada classificador SVM obtém previsões independentes para cada classe da hierarquia, que são então transformadas em estimativas de probabilidade (pertinência) de acordo com (Zadrozny e Elkan, 2002). As estimativas de probabilidades de cada exemplo sendo classificado são então fornecidas como entrada para um algoritmo chamado *UniSmooth*, proposto pelos autores. O algoritmo proposto faz uma suavização dos valores de probabilidade iniciais, com a finalidade de garantir a consistência das previsões de acordo com a hierarquia de classes sendo investigada. O algoritmo *UniSmooth* utiliza o conceito de *peak-node*, no qual cada caminho predito em uma hierarquia de classes possui um nó cujo valor de pertinência seja maior que os valores de pertinência de todos os seus descendentes e ancestrais. Experimentos em conjuntos de dados de documentos mostraram que o método pode obter bons resultados.

Uma combinação de classificadores (*ensemble*) utilizando a abordagem LCN foi proposta no trabalho de Valentini (2009), em que um classificador base é associado a cada classe da hierarquia. O método, chamado de *True Path Rule Ensemble* (TPR), foi aplicado a conjuntos de dados de genes anotados de acordo com o esquema FunCat (Ruepp et al., 2004) desenvolvido pelo MIPS (Mewes et al., 2002). Os classificadores base são treinados para tornarem-se especializados na classificação de uma classe específica da hierarquia. Para isso, cada classificador treinado estima a probabilidade local $\hat{p}_j(\mathbf{x}_i)$ de um determinado exemplo $\mathbf{x}_i$ pertencer a uma classe c_j . A fase de combinação estima a probabilidade consensual global $p_j(\mathbf{x}_i)$.

Nos trabalhos de Valentini e Re (2009) e Valentini (2011), os autores modificaram o trabalho de Valentini (2009) com o objetivo de modular a relação entre a previsão de uma classe e a previsão de seus descendentes. Para isso, os autores utilizaram um peso nas classes pais denotado por w_p , com $0 \leq w_p \leq 1$, tal que se $w_p = 1$, a decisão da classificação de um nó i depende somente do classificador local nesse nó. Para valores de w_p diferentes de 1, a decisão da previsão naquele nó é dividida proporcionalmente entre o classificador local no nó e o conjunto de todos os seus classificadores filhos. Os resultados experimentais em dados biológicos mostraram bons resultados em comparação com o mesmo método sem a utilização dos pesos w_p . O novo método foi chamado de *Weighted True Path Rule* (TPR-W).

No trabalho de Chen e Hu (2010), é proposto um método baseado na abordagem LCN utilizando classificadores SVM binários, que estende a técnica de combinação de classificadores proposta em (Valentini, 2011). Enquanto em (Valentini, 2011), cada classificador associado aos nós filhos contribui igualmente na estimativa das probabilidades locais ($\hat{p}_j(\mathbf{x}_i)$), em (Chen e Hu, 2010), classificadores que têm como saída altos valores de probabilidade para exemplos negativos de uma dada classe contribuem menos para o cálculo das probabilidades locais. Além de melhoramentos na técnica de *ensemble*, os autores propõem uma técnica de *over-sampling* hierárquica, baseada na técnica *Synthetic Minority Over-sampling Technique* (SMOTE) (Chawla et al., 2002), com a finalidade de minimizar os efeitos do grande desbalanceamento dos conjuntos de dados, e com isso melhorar o desempenho dos classificadores SVM individuais. Os classificadores SVM utilizados têm como saída uma probabilidade *a posteriori* estimada de acordo com a técnica descrita em (Platt, 1999).

Os métodos *HBayes* (Cesa-Bianchi et al., 2006) e *True Path Rule Ensemble* (TPR) (Valen-

tini, 2009), e seus variantes *HBayes-CS* (Cesa-Bianchi e Valentini, 2010) e *Weighted True Path Rule* (TPR-W) (Valentini e Re, 2009), foram comparados no trabalho de Re e Valentini (2010a). Os experimentos sugeriram que, embora os resultados obtidos por todos os métodos fossem semelhantes, as variações obtiveram resultados um pouco melhores do que suas versões originais.

Outra estratégia local foi proposta por Kiritchenko et al. (2004), e aplicada a conjuntos de dados compostos por artigos do repositório MedLine (MedLine database, 2008) associados a códigos da GO. A fase de treinamento segue a abordagem LCPN e, durante o processo de classificação *top-down*, cada classificador tem como saída um valor real representando a probabilidade de um exemplo de entrada pertencer a uma classe c_j . Somente as classes cujas probabilidades são maiores que um dado limiar são atribuídas ao exemplo.

Outro método baseado na abordagem LCPN, chamado TreeBoost.MH, foi proposto por Esuli et al. (2006) e Esuli et al. (2008), e trata-se de uma variação hierárquica do método AdaBoost.MH (Schapire e Singer, 1999) para ser aplicado em hierarquias estruturadas como árvores. O método foi aplicado à classificação de documentos, utilizando uma versão hierárquica do corpus Reuters-21578, gerado em (Toutanova et al., 2001), e o corpus *Reuters Corpus Volume 1* - versão 2 (RCV1-v2). O método TreeBoost.MH foi definido como um algoritmo recursivo que usa o método AdaBoost.MH como algoritmo base, e é chamado recursivamente nos nós internos da estrutura hierárquica.

Em (Cerri e Carvalho, 2010b) foram propostos dois novos métodos de classificação hierárquica multirrótulo seguindo a abordagem LCPN, baseados em dois métodos multirrótulo não hierárquicos existentes na literatura. O primeiro deles é chamado *Hierarchical Multi-Label Classification with Label-Powerset* (HMC-LP) e é baseado no método *Label-Powerset*, utilizado em (Boutell et al., 2004) e (Tsoumakas e Vlahavas, 2007). O segundo método é chamado *Hierarchical Multi-Label Classification with Cross-Training* (HMC-CT), baseado no método *Cross-Training* (Shen et al., 2003). Em cada nível da hierarquia de classes, o método HMC-LP combina as classes atribuídas a um exemplo em uma nova e única classe, transformando o problema hierárquico multirrótulo original em um problema hierárquico simples-rótulo. O método HMC-CT utiliza uma estratégia de decomposição de classes, transformando o problema hierárquico multirrótulo original em um conjunto de problemas hierárquicos simples-rótulo. Foi utilizado o algoritmo SVM como classificador base, e experimentos em conjuntos de dados biológicos mostraram bons resultados. Recentemente, o trabalho de Cerri et al. (2011) estendeu o trabalho de Cerri e Carvalho (2010b), utilizando diferentes algoritmos convencionais de classificação como classificadores base, como o algoritmo C4.5 (Quinlan, 1993), Redes Bayesianas (Friedman et al., 1997), o algoritmo KNN (Aha et al., 1991), e o algoritmo de geração de regras de decisão *Ripper* (Cohen, 1995).

No trabalho de Díez et al. (2010), foi proposto um método baseado na abordagem LCN utilizando classificadores SVM binários. O método proposto utiliza uma estratégia de aprendizado *bottom-up*, em que o treinamento de um classificador associado a uma classe c_j acontece após o treinamento de todos os classificadores associados às subclasses da classe c_j . Durante o processo *bottom-up*, uma função de custo é utilizada para o treinamento dos classificadores

associados aos nós internos da hierarquia. Essa função de custo é utilizada para calcular o custo de cada exemplo atribuído às classes filhas da classe associada ao classificador sendo treinado. Assim, é possível identificar os exemplos mais relevantes para o processo de treinamento, e atribuir pesos aos exemplos de acordo com suas relevâncias. Após finalizada a fase de treinamento, a classificação de novos exemplos é obtida por meio de um processo *top-down*. Experimentos em conjuntos de patentes WIPO-alpha (WIPO, 2001), e de documentos (Lewis et al., 2004), mostraram que o método proposto pode obter bons resultados.

A abordagem LCN também foi utilizada em (Chen et al., 2010), associando classificadores SVM binários a cada nó da hierarquia de classes. Nesse trabalho, os autores propuseram duas técnicas para contornar o problema do bloqueio presente na abordagem local. Assim como em (Sun et al., 2004), a técnica proposta diminui os valores de limiares dos classificadores conforme o nível da hierarquia torna-se mais profundo. Duas alternativas foram investigadas. A primeira considera a quantidade de exemplos atribuídos a uma classe e às superclasses diretas dessa classe, de maneira a verificar quantos exemplos são transferidos de pai para filho na hierarquia. A segunda utiliza um procedimento parecido com o utilizado em (Clare e King, 2003), com a finalidade de obter a informação de uma dada classe. Quanto mais profunda é uma classe, mais informativa ela é, e portanto um menor valor de limiar deve ser atribuído ao seu classificador associado. Experimentos utilizando conjuntos de dados biológicos mostraram que o método local, utilizado com as estratégias de diminuição dos limiares, obteve melhores resultados do que o mesmo método aplicado com limiares fixos.

No trabalho de Bi e Kwok (2011) foi proposto um método que utiliza transformação de problemas hierárquicos multirrótulo em problemas simples-rótulo. O método foi aplicado em hierarquias estruturadas tanto como árvores quanto DAGs. Inicialmente, as classes dos exemplos são projetadas para um espaço $\mathcal{R}^m$ de menor dimensão, utilizando *Kernel Principal Component Analysis*. Em seguida, m regressores (*ridge regression* (Hoerl e Kennard, 1970)) são treinados, e as predições para um exemplo de teste $\mathbf{x}$ são obtidas utilizando os m modelos induzidos. As predições são então mapeadas para o espaço original. Para preservar os relacionamentos hierárquicos durante o processo de projeção dos vetores de classes, as restrições hierárquicas foram incorporadas à função de *Kernel* utilizada, conforme (Tsochantaridis et al., 2005). Para o processo de predição, foram utilizados dois métodos inspirados em um algoritmo chamado *Condensing Sort and Select Algorithm* (CSSA) (Baraniuk e Jones, 1994), proposto para encontrar uma subárvore aproximada ótima em uma árvore. Esse mesmo método foi estendido para encontrar um subgrafo ótimo em um grafo. Experimentos comparativos com o método Clus-HMC (Vens et al., 2008) sugeriram que o método proposto obteve resultados comparativos, porém o mesmo não gera regras de classificação. O método pode ser considerado como pertencente à abordagem LCN, já que associa modelos de regressão à cada classe resultante do processo de projeção das classes para um espaço de menor dimensão.

Em (Cesa-Bianchi et al., 2011) foi investigada a sinergia entre diferentes estratégias relacionadas à tarefa de predição de funções de genes. Para isso, os autores integraram ferramentas de fusão de dados baseadas em *kernel* e algoritmos de *ensemble* (Re e Valentini, 2010b), com

algoritmos de classificação hierárquica multirrótulo sensíveis ao custo de classificação (Cesa-Bianchi e Valentini, 2010; Valentini, 2011). Os autores definem como sinergia a melhora, considerando alguma medida de avaliação, no desempenho de classificação devido à utilização concorrente de estratégias de aprendizado. A sinergia é detectada sempre que a ação combinada de duas estratégias faz com que o desempenho na classificação seja melhor do que a média dos desempenhos das duas estratégias utilizadas separadamente (Cesa-Bianchi et al., 2011). Os autores concluíram experimentalmente que a integração de algoritmos de classificação hierárquica multirrótulo, técnicas de fusão de dados que capturam diferentes características funcionais de genes, e métodos sensíveis ao custo de classificação, levam a melhora no desempenho preditivo. Os autores, no entanto, não utilizaram em seus experimentos outros métodos de classificação presentes na literatura. Utilizaram apenas algoritmos que foram propostos pelos próprios autores, que seguem a abordagem LCN.

Há também trabalhos propondo métodos que utilizam a abordagem global (GC). O trabalho de Blockeel et al. (2002) apresentou um algoritmo chamado Clus-HMC, que gera uma única árvore de decisão para toda a hierarquia de classes, e é baseado na noção de *Predictive Clustering Trees* (PCT) (Blockeel et al., 1998), em que árvores de decisão são vistas como uma hierarquia de grupos (*clusters*). O mesmo algoritmo foi também utilizado nos trabalhos de Struyf et al. (2005) e Blockeel et al. (2006). O algoritmo para indução de árvores de decisão é essencialmente um algoritmo *Top-Down Induction of Decision Trees* (TDIDT). A ideia geral é particionar recursivamente o conjunto de dados em grupos, de maneira que a variância intra-grupos seja minimizada. A heurística utilizada para selecionar os testes que são incluídos como nós da árvore é baseada nessa variância. Na árvore de decisão produzida por esse algoritmo, os nós folhas não contêm classes, e cada nó, assim como os nós folhas, corresponde a um grupo. Para calcular a distância entre dois grupos, é utilizada uma função que calcula o valor da distância entre os valores dos protótipos (*prototype*) desses dois grupos. O valor do protótipo de um grupo de exemplos pode ser considerado como uma descrição parcial dos exemplos. Na árvore de decisão final, cada nó folha conterá o valor do protótipo dos exemplos cobertos por aquela folha. Experimentos utilizando conjuntos de dados de documentos e biológicos mostraram bons resultados na classificação.

O trabalho de Vens et al. (2008) comparou três métodos baseados no conceito de PCT para a resolução de problemas hierárquicos multirrótulo relacionados à genômica funcional. Os autores utilizaram o método Clus-HMC (Blockeel et al., 2002), baseado na abordagem global, que induz uma única árvore de decisão e faz previsões considerando todas as classes de uma única vez, e compararam seu desempenho com os desempenhos de dois métodos locais, chamados Clus-SC e Clus-HSC. O método Clus-SC define uma tarefa de classificação hierárquica simples-rótulo independente para cada classe, ignorando relacionamentos hierárquicos. O método Clus-HSC explora os relacionamentos hierárquicos para induzir uma árvore de decisão para cada classe da hierarquia. Os autores também aplicaram os métodos a hierarquias estruturadas como DAGs, discutindo as questões que surgem quando é utilizada esse tipo de estrutura hierárquica e as modificações necessárias nos algoritmos para tornar possível a aplicação dos

mesmos nessas estruturas.

Enquanto no trabalho de Vens et al. (2008) os autores utilizaram a distância Euclidiana como medida para calcular a similaridade ou dissimilaridade entre os exemplos na árvore de decisão, o trabalho de Aleksovski et al. (2009) investigou a utilização de outras medidas de distância, chamadas distância Jaccard, distância SimGIC e distância ImageClef. Os autores concluíram que não houve diferenças estatisticamente significantes entre as diferentes medidas de distância utilizadas.

Nos trabalhos de Kiritchenko et al. (2005) e Kiritchenko et al. (2006), foi utilizado um método baseado na abordagem global para a classificação de genes da GO. Assim como feito no trabalho de Kiritchenko et al. (2004), a classificação dos genes foi baseada na classificação de documentos do repositório MedLine que descrevem esses genes. O método consiste em expandir os conjuntos de classes dos exemplos de maneira que contenham as classes ancestrais de suas classes, e depois aplicar o método AdaBoost.MH no novo conjunto de dados criado. Após essa fase, é feita uma correção de predições inconsistentes que podem ter ocorrido. A comparação com outros métodos não hierárquicos e hierárquicos baseados na abordagem local, que usam o algoritmo AdaBoost, mostrou bons resultados.

O trabalho de Rousu et al. (2006), investigou a utilização de um algoritmo baseado em funções de *kernel* para a tarefa de classificação hierárquica multirrótulo. O método foi aplicado a problemas de classificação de textos e dados biológicos (*Reuters Corpus Volume 1*, patentes WIPO-alpha (WIPO, 2001) e enzimas (Webb et al., 1992)). O método de classificação é baseado em uma variação de um *framework* chamado *Maximum Margin Markov Network* (Taskar et al., 2003; Tsochantaridis et al., 2005), em que a estrutura hierárquica é representada como uma árvore de Markov. Para o processo de aprendizado, os autores definiram uma função que mapeia as características dos exemplos para os espaços de entrada e saída. No contexto de classificação hierárquica multirrótulo, o espaço de saída pode ser definido como todas as possíveis subárvores ou subgrafos da hierarquia de classes.

Outro método baseado na abordagem GC foi proposto por Clare e King (2003). Esse método global, chamado HMC4.5, é baseado em indução de árvores de decisão, e foi aplicado à classificação de funções de genes do organismo *Saccharomyces cerevisiae*. O método é uma variação do algoritmo de indução de árvores de decisão C4.5, em que foram feitas modificações na fórmula do cálculo da entropia de classes. No algoritmo C4.5 original, a entropia é utilizada para decidir a melhor divisão dos dados na árvore de decisão, ou seja, o melhor atributo a ser colocado como nó da árvore. A modificação dos autores utiliza a soma das entropias de todas as classes para escolher o melhor atributo a ser utilizado para rotular um nó interno da árvore de decisão. Além disso, a informação sobre o tamanho da subárvore enraizada em uma determinada classe também é utilizada na fórmula da entropia.

A estratégia global também foi utilizada no trabalho de Cai e Hofmann (2007), em que um novo método foi proposto e aplicado à tarefa de classificação de documentos, utilizando o conjunto de dados de patentes WIPO-alpha. O método é uma variação hierárquica multirrótulo do algoritmo SVM, em que os autores generalizaram a formulação do problema multiclasse

(Crammer e Singer, 2002) para uma formulação multirrótulo similar a feita por Elisseeff e Weston (2001). Seguindo a sugestão de Cai e Hofmann (2004), foi utilizada uma função F que é linear em alguma representação conjunta de características Φ dos exemplos de entrada e das classes, denotada por $F(\mathbf{x}, c; \mathbf{w}) \equiv \langle \mathbf{w}, \Phi(\mathbf{x}, c) \rangle$, com $\mathbf{w}$ sendo um vetor de pesos, $\mathbf{x}$ um exemplo de entrada e c uma classe. A margem de separação de $\mathbf{w}$ correspondente ao i -ésimo exemplo pode ser aproximada como $\gamma_i(\mathbf{w}) \equiv \min_{c \in C'_i, \bar{c} \in \bar{C}'_i} \langle \Phi(\mathbf{x}_i, c) - \Phi(\mathbf{x}_i, \bar{c}), \mathbf{w} \rangle$, em que C'_i denota um dado conjunto verdadeiro de classes e $\bar{C}'_i$ seu complemento. A maximização da margem de separação sobre todo o conjunto de exemplos é formulada como $\max_{\mathbf{w}: \|\mathbf{w}\|=1} \min_i \gamma_i(\mathbf{w})$.

Em (Schietgat et al., 2010) foi utilizada uma técnica de *ensemble* para combinar as árvores de decisão induzidas pelo método Clus-HMC (Vens et al., 2008). O método proposto foi chamado de Clus-HMC-ENS. Os autores utilizaram a técnica de combinação *Bagging* (Breiman, 1996), que constrói diferentes classificadores fazendo replicações (*bootstrap*) do conjunto de dados de treinamento, e utilizando cada uma dessas replicações para a indução de um classificador. Experimentos em conjuntos de dados biológicos mostraram melhoras significativas nos resultados obtidos. Em (Dimitrovski et al., 2009, 2010, 2011), a combinação de classificadores baseados em PCT também foi investigada, utilizando *Bagging* e *Random Forests* (Breiman, 2001), apresentando bons resultados preditivos.

Em (Sangsuriyun et al., 2010), é proposto um método baseado na abordagem global, chamado *Hierarchical Multi-Label Associative Classification* (HMAC). O algoritmo proposto utiliza regras chamadas de regras negativas, que também consideram importante a ausência de uma determinada característica (atributo) para a classificação de um exemplo. Além dos atributos, o método também considera regras que predizem um conjunto negativo de classes, ou seja, dizem que um exemplo não pertence a determinado conjunto de classes. O método é dividido em três fases. A primeira fase gera regras que então sofrem um processo de poda na fase seguinte. O processo de poda é feito para simplificação e eliminação de redundâncias nas regras geradas. Uma terceira fase então constrói um conjunto final de regras, formado pelas melhores regras do conjunto inicial. O método proposto foi aplicado a conjuntos de dados biológicos, apresentando resultados satisfatórios.

Outro método baseado na abordagem global foi proposto no trabalho de Pugelj e Džeroski (2011). O nome do método é KNN-SP, e trata-se do algoritmo KNN com uma modificação para classificação hierárquica multirrótulo. A modificação consiste em calcular o protótipo dos k -vizinhos mais próximos de um exemplo a ser classificado. Esse protótipo será a predição final do exemplo, assim como feito em (Vens et al., 2008). O método foi implementado utilizando o framework utilizado para a implementação do Clus-HMC (Vens et al., 2008) e apresentou resultados competitivos quando comparado com esse mesmo método.

No trabalho de Wang et al. (2011), foi proposto um método global baseado no algoritmo KNN, chamado de *Enhanced K-Nearest Neighbour* (EKNN). O método utiliza uma nova estratégia para a aplicação de limiares no momento da classificação multirrótulo. Essa nova estratégia combina duas estratégias chamadas de *Score-cut* e *Rank-cut*. A primeira prediz uma classe se sua confiança for maior que um dado limiar, enquanto a segunda prediz as n classes com

maiores valores de confiança. O método foi aplicado à tarefa de classificação de textos.

Um método específico para problemas de classificação mandatória em nós-folha foi proposto em (Bi e Kwok, 2012). O método segue a abordagem global e utiliza a maximização da probabilidade a posteriori entre todas as classes do problema. Dessa forma, um exemplo é classificado no conjunto de nós da hierarquia que possui a maior probabilidade a posteriori entre todos os nós. O método, chamado *MAandatory leaf node prediction on Structures* (MAS), foi aplicado a conjuntos de dados de textos e biológicos, com hierarquias estruturadas como árvores e grafos.

3.4.2 Abordagens Bioinspiradas

Em (Cerri e Carvalho, 2010a), Redes Neurais Artificiais (RNA) foram utilizadas como classificadores base no método de classificação hierárquica multirrótulo HMC-LP (Cerri e Carvalho, 2010b; Cerri et al., 2011), baseado na abordagem LCPN. O método proposto é uma adaptação do método de classificação multirrótulo não hierárquico chamado *Label-Powerset*, utilizado em Boutell et al. (2004) e Tsoumakas e Vlahavas (2007). Em cada nível da hierarquia de classes, o método HMC-LP combina as classes atribuídas a um exemplo em uma nova e única classe, transformando o problema hierárquico multirrótulo original em um problema hierárquico simples rótulo. Após essa simplificação, a abordagem local é utilizada para treinamento e teste do método, utilizando RNA como classificadores base.

No trabalho de Cai e Hofmann (2007) foi desenvolvido um método baseado na abordagem global utilizando computação bioinspirada. Os autores propuseram um Perceptron hierárquico utilizando a regra de aprendizado mínima sobreposição (*minimum overlap* (Minover)) (Krauth e Mezard, 1987). O método utiliza os exemplos que mais violam a margem de separação desejada para atualizar o hiperplano de separação. Somente os vetores de pesos dos nós que são predecessores de $c \in C'$ ou $\bar{c} \in \bar{C}'$ (C' denota um dado conjunto verdadeiro de classes e $\bar{C}'$ seu complemento), mas não ambos, são atualizados. Os outros nós permanecem inalterados.

Um perceptron hierárquico foi desenvolvido no trabalho de Woolam e Khan (2008), estendendo o algoritmo *Hieron* proposto no trabalho de Dekel et al. (2004). O método proposto, chamado de *MultiHieron*, preservou as características do algoritmo original, mas introduziu modificações para garantir que as predições sejam multirrótulo. Modificações foram feitas para garantir o cálculo das margens de separação, e também foi introduzida uma função de erro mais adequada para problemas multirrótulo. Experimentos em conjuntos de dados de texto mostraram bons resultados na acurácia preditiva.

No trabalho de Alves et al. (2008), também foi proposto um método baseado na abordagem global utilizando Sistemas Imunes Artificiais (SIA) (de Castro e Timmis, 2002). O método foi chamado *Multi-label Hierarchical Classification with an Artificial Immune System* (MHC-AIS) e utilizado na tarefa de predição de proteínas descritas na GO. Segundo os autores, o algoritmo proposto é capaz de encontrar um conjunto de regras que sejam tanto hierárquicas quanto multirrótulo, tal que uma única regra de classificação seja capaz de atribuir mais de um caminho de uma estrutura hierárquica a um exemplo. O algoritmo de treinamento é dividido em

dois procedimentos básicos chamados *Sequential Covering* (SC) e *Rule Evolution* (RE). Esses procedimentos produzem regras de classificação candidatas, sendo cada regra composta de duas partes: um antecedente (*IF part*), representado por um vetor de condições atributo-valor, e um conseqüente (*THEN part*), representado por um conjunto de classes preditas. O procedimento SC iterativamente chama o procedimento RE até que todos (ou quase todos) os “antígenos” (exemplos de treinamento) sejam cobertos pelas regras descobertas. O procedimento RE evolui “anticorpos” (regras de classificação) que são utilizadas para classificar os antígenos. O melhor anticorpo é adicionado ao conjunto de regras descobertas.

Em (Alves et al., 2010) foram adicionados alguns procedimentos ao algoritmo proposto em (Alves et al., 2008), a fim de melhorar seu desempenho. O primeiro procedimento consiste em recalcular o *fitness* de todas as regras considerando todo o conjunto de exemplos de treinamento e, remover dos conseqüentes das regras, todas as classes que apresentarem um valor de *fitness* menor que um dado limiar. Outra modificação foi a utilização de procedimentos de poda e busca local na melhor regra encontrada pelo procedimento *Rule Evolution*, de maneira a obter maior simplicidade e precisão. Experimentos em conjuntos de dados da GO mostraram bons resultados.

Em (Otero et al., 2010), foi estendido o método global chamado *hAnt-Miner* (Otero et al., 2009), um método de inteligência de enxames baseado em Otimização por Colônias de Formigas (OCF) (Dorigo et al., 1991; Dorigo, 1992) para a classificação hierárquica simples-rótulo. O método *hAnt-Miner* tem o objetivo de descobrir regras hierárquicas de classificação na forma *SE antecedente ENTÃO conseqüente*. A construção de uma regra é dividida em duas colônias de formigas, uma colônia para criar o antecedente de uma regra e outra para criar o conseqüente, que trabalham de forma cooperativa. Basicamente, um processo de cobertura sequencial de exemplos é empregado para criar regras de classificação que cubram todos, ou quase todos, os exemplos de treinamento. O algoritmo é iniciado com uma lista vazia de regras, e uma nova regra é adicionada à lista enquanto o número de exemplos não cobertos por nenhuma regra for maior que um dado parâmetro. A cada iteração, uma regra é construída por meio do pareamento de uma formiga da colônia responsável pela criação dos antecedentes das regras, com uma formiga da colônia responsável pela criação dos conseqüentes das regras. O novo método proposto, chamado *hmAnt-Miner*, difere do original nos seguintes aspectos: (i) o conseqüente de uma regra é obtido utilizando um procedimento determinístico baseado nos exemplos cobertos pela regra, e não mais utilizando uma colônia específica. Isso fez com que as regras geradas agora fossem hierárquicas multirrótulo; (ii) a heurística utilizada agora é baseada na distância Euclidiana entre os conseqüentes das regras, em que cada conseqüente é representado por um vetor de valores de pertinência de classes; (iii) a qualidade de uma regra é avaliada utilizando uma medida baseada em distância; e (iv) o procedimento de poda de uma regra não é aplicado ao conseqüente da regra. Ao invés disso, o conseqüente é recalculado quando seu antecedente é modificado durante a poda.

No trabalho de Brucker et al. (2010), foi investigado o problema de extração de hierarquias a partir dos resultados obtidos por um classificador multirrótulo não hierárquico. Nesse traba-

lho, supõe-se que a hierarquia seja desconhecida e que tenha que ser inferida automaticamente a partir de predições multirrótulo. O método proposto utiliza um classificador multirrótulo chamado ML-ARAM (Sapozhnikova, 2009), pertencente à família de redes neurais *Adaptive Resonance Theory* (ART) (Carpenter e Grossberg, 1987). A rede ML-ARAM é uma extensão multirrótulo do método *Adaptive Resonance Associative Map* (ARAM) (Tan, 1995). Associado ao classificador multirrótulo, é utilizado um algoritmo de aprendizado de regras e um algoritmo para extração de hierarquias. Outros dois métodos de classificação multirrótulo também foram utilizados no lugar do método ML-ARAM, chamados BoosTexter (Schapire e Singer, 2000) e ML-kNN (Zhang e Zhou, 2007). Basicamente, após a classificação multirrótulo, o método cria regras envolvendo todas as classes preditas. O procedimento de construção de regras assegura que, quanto mais frequentemente uma classe aparecer nos consequentes das regras, mais alto será o seu nível na hierarquia. Experimentos em conjuntos de dados de textos mostraram bons resultados na predição multirrótulo, e também as hierarquias geradas foram comparadas com as hierarquias verdadeiras dos conjuntos de dados utilizados, mostrando consistência das hierarquias encontradas. O método, no entanto, não é considerado aqui como um método de classificação hierárquica multirrótulo, pois não faz considerações sobre a hierarquia de classes nas fases de treinamento e teste. O método é visto essencialmente como um método de classificação multirrótulo não hierárquica, sendo que a hierarquia de classes é inferida apenas após a predição multirrótulo final.

Em (Borges e Nievola, 2012), foi desenvolvido um método global utilizando uma rede neural competitiva. A rede neural consiste de uma camada de entrada, formada pelos atributos dos exemplos, e de uma camada de saída, na qual cada neurônio corresponde a um nó da hierarquia. O processo de treinamento da rede consiste de três etapas: competição, cooperação e adaptação. Na etapa de cooperação, é calculada, para cada exemplo de treino, a distância entre esse exemplo e todos os nós da hierarquia. Os neurônios que possuem a menor distância são então considerados os neurônios vencedores, e irão influenciar todos os seus neurônios ancestrais (neurônios correspondentes aos nós ancestrais dos nós considerados vencedores). A fase de cooperação consiste então em encontrar todos os neurônios ancestrais dos neurônios vencedores. Na fase de adaptação, os pesos sinápticos conectando os atributos de entrada aos neurônios vencedores, e também seus ancestrais, são ajustados. Esse ajuste consiste em aproximar ou afastar os vetores de pesos dos vetores de atributos dos exemplos. Se a classe associada a um neurônio vencedor for igual à classe do exemplo em questão, os pesos são ajustados para aproximar os vetores. Caso contrário os pesos serão ajustados de maneira a distanciar os vetores. O método, chamado Multi-Label Hierarchical Classification Using a Competitive Neural Network (MHC-CNN) foi aplicado a conjuntos de dados biológicos com hierarquias estruturadas como DAGs.

A tabela 3.1 apresenta os trabalhos revisados nesta seção, categorizados de acordo com a taxonomia proposta em (Silla e Freitas, 2010).

Tabela 3.1: Categorização dos métodos de acordo com a taxonomia proposta por (Silla e Freitas, 2010).

< $\Delta, \Xi, \Omega, \Theta$ >	Lista de trabalhos
<MPP,NMLNP,D,LCN>	Nguyen et al. (2005); Barutcuoglu et al. (2006); Barutcuoglu e DeCoro (2006); DeCoro et al. (2007); Guan et al. (2008); Jin et al. (2008); Punera e Rajan (2009); Chen et al. (2010); Bi e Kwok (2011)
<MPP,NMLNP,T,LCN>	Dumais e Chen (2000); Sun e Lim (2001); Sun et al. (2003, 2004); Cesa-Bianchi et al. (2006); Valentini (2009, 2011); Valentini e Re (2009); Cesa-Bianchi e Valentini (2010); Bennett e Nguyen (2009); Mayne e Perry (2009); Re e Valentini (2010a); Chen e Hu (2010); Díez et al. (2010); Cesa-Bianchi et al. (2011)
<MPP,NMLNP,D,LCPN>	Kiritchenko et al. (2004)
<MPP,MLNP,T,LCPN>	Esuli et al. (2006, 2008); Cerri e Carvalho (2010b,a); Cerri et al. (2011)
<MPP,NMLNP,T,GC>	Rousu et al. (2006); Clare e King (2003); Cai e Hofmann (2004, 2007); Woolam e Khan (2008)
<MPP,NMLNP,D,GC>	Blockeel et al. (2002); Struyf et al. (2005); Kiritchenko et al. (2005, 2006); Blockeel et al. (2006); Cai e Hofmann (2007); Alves et al. (2008, 2010); Vens et al. (2008); Aleksovski et al. (2009); Dimitrovski et al. (2009, 2010, 2011); Schietgat et al. (2010); Otero et al. (2010); Sangsuriyun et al. (2010); Pugelj e Džeroski (2011); Wang et al. (2011); Bi e Kwok (2012); Borges e Nievola (2012)

3.5 Revisão de Medidas de Avaliação

Medidas de avaliação para problemas de classificação convencionais (não hierárquicos e simples-rótulo) são geralmente inadequadas para problemas de classificação hierárquica multirrótulo. Além de não considerarem as estruturas hierárquicas dos problemas e o fato de um exemplo poder pertencer a mais de uma classe simultaneamente, medidas convencionais de avaliação ignoram que a dificuldade da tarefa de classificação aumenta conforme aumenta a profundidade das classes a serem preditas. Em problemas de classificação hierárquica, classes mais específicas (mais profundas) são geralmente mais difíceis de serem preditas do que classes mais genéricas (mais próximas da raiz), e as medidas convencionais não levam isso em consideração porque assumem que os custos atribuídos aos erros de classificação são independentes das posições das classes reais e preditas na hierarquia. Além disso, na classificação multirrótulo, essas medidas não consideram que um exemplo pode ser atribuído a apenas um subconjunto de suas classes verdadeiras.

Como alternativas para medidas de avaliação convencionais, medidas específicas para problemas hierárquicos, multirrótulo e hierárquicos multirrótulo têm sido propostas. Esta seção revisa medidas de classificação hierárquica multirrótulo, que podem ser separadas em três grupos principais: (i) medidas baseadas em hierarquia, (ii) medidas baseadas em distância e (iii) medidas baseadas em semântica. Enquanto as medidas baseadas em hierarquia consideram a estrutura hierárquica dos problemas, medidas baseadas em distância e semântica consideram, respectivamente, a distância entre as classes preditas e verdadeiras e os relacionamentos semânticos entre as classes.

Apesar de muitos trabalhos da literatura avaliarem o desempenho de classificadores hierárquicos multirrótulo, parece não haver consenso sobre qual medida é mais apropriada para ser

utilizada em um tipo de conjunto de dados ou método de classificação específico, o que faz com que surjam algumas questões como: o balanceamento da hierarquia de classes ou a porcentagem de exemplos multirrótulo em um conjunto de dados favorece uma medida em detrimento de outras? Abordagens locais e globais devem ser avaliadas com medidas específicas? Assim, além de revisar as principais medidas de avaliação propostas na literatura, esta seção também discute os prós e contras dessas diferentes medidas.

3.5.1 Medidas de Avaliação Baseadas em Hierarquia

Medidas de avaliação baseadas em hierarquia consideram as classes ancestrais e descendentes das classes preditas no momento da avaliação. Duas medidas baseadas nesse conceito são apresentadas: Precisão Hierárquica (PH), Revocação Hierárquica (RH), e *Hierarchical Loss Function* (H-Loss). Adicionalmente, outras medidas de avaliação baseadas em hierarquia encontradas na literatura são brevemente apresentadas.

Precisão e Revocação Hierárquicas

Em Kiritchenko et al. (2004), duas medidas de avaliação baseadas nas medidas convencionais de Precisão e Revocação foram propostas, considerando os relacionamentos hierárquicos entre as classes. Essas duas medidas são chamadas Precisão Hierárquica (PH) e Revocação Hierárquica (RH), e foram formalmente definidas em (Kiritchenko et al., 2005). Essas mesmas medidas de avaliação também foram utilizadas em (Eisner et al., 2005), (Verspoor et al., 2006) e (Jin et al., 2008).

Essas medidas consideram que um exemplo pertence não apenas a uma classe, mas também a todas as classes ancestrais dessa classe na estrutura hierárquica. Assim, dada uma tupla $(\mathbf{x}_i, C_i)$, com $\mathbf{x}_i$ pertencendo ao espaço $\mathbf{X}$ de exemplos, C_i o conjunto de classes preditas do exemplo $\mathbf{x}_i$, e C'_i o conjunto de classes verdadeiras do exemplo $\mathbf{x}_i$, os conjuntos C_i e C'_i podem ser estendidos para conterem suas correspondentes classes ancestrais da seguinte maneira: $\widehat{C}_i = \bigcup_{c_k \in C_i} \text{Ancestrais}(c_k)$ e $\widehat{C}'_i = \bigcup_{c_l \in C'_i} \text{Ancestrais}(c_l)$, em que $\text{Ancestrais}(c_k)$ denota o conjunto de ancestrais da classe c_k .

As Equações 3.3 e 3.4 apresentam os cálculos da PH e da RH.

$$PH = \frac{\sum_i |\widehat{C}_i \cap \widehat{C}'_i|}{\sum_i |\widehat{C}_i|} \quad (3.3)$$

$$RH = \frac{\sum_i |\widehat{C}_i \cap \widehat{C}'_i|}{\sum_i |\widehat{C}'_i|} \quad (3.4)$$

Essas medidas contam o número de classes corretamente preditas, e também o número de ancestrais corretamente preditos dessas classes, assumindo que exemplos também pertencem às classes ancestrais de suas classes (Kiritchenko et al., 2005). A Figura 3.4 apresenta um exemplo do cálculo dessas medidas, em que os círculos sólidos representam as classes folhas verdadeiras de um exemplo, e os círculos em negrito com uma seta representam as classes mais profundas

que foram preditas para um exemplo.

Figura 3.4: Representação gráfica das medidas de Precisão e Revocação Hierárquicas. Adaptado de (Kiritchenko et al., 2004).

Como pode ser observado na figura, todos os nós no caminho do nó raiz até a classe predita de um exemplo estão em negrito, porque as classes ancestrais das classes preditas também são atribuídas ao exemplo. As arestas no caminho do nó raiz até o nó que representa a classe real de um exemplo também são mostradas em negrito. Os valores de **PH** e **RH** para três diferentes predições também são mostrados na figura.

As medidas de Precisão Hierárquica e Revocação Hierárquica utilizadas isoladamente não são adequadas para a avaliação de classificadores (Sebastiani, 2002). Sendo assim, as medidas **PH** e **RH** são combinadas em uma extensão hierárquica da medida-F, chamada *Hierarchical- F_β* , apresentada na Equação 3.5. Na equação, β representa a importância atribuída aos valores de **PH** e **RH**. Conforme o valor de β é aumentado, o peso atribuído ao valor de **RH** também é aumentado. Por outro lado, se o valor de β for diminuído, o valor de **PH** é aumentado.

$$\text{Hierarchical} - F_\beta = \frac{(\beta^2 + 1) \times PH \times RH}{\beta^2 \times PH + RH} \quad (3.5)$$

Hierarchical Loss Function

No trabalho de Cesa-Bianchi et al. (2006), uma medida de avaliação chamada *Hierarchical Loss Function* (**H-Loss**) foi proposta. Essa medida é baseada no conceito de que quando um erro de classificação ocorre em uma classe da hierarquia, penalizações adicionais não devem ser dadas a erros cometidos na subárvore dessa classe, ou seja, se um erro ocorre em uma classe c_j , erros adicionais cometidos na subárvore enraizada na classe c_j não são importantes. Como exemplo, se um classificador erroneamente classifica um exemplo como pertencendo à classe “esportes”, esse classificador não deve ser penalizado novamente por classificar erroneamente o mesmo documento na subclasse “futebol”.

Considere que o conjunto de classes atribuído a um dado exemplo $\mathbf{x}_i$ é qualquer subconjunto do conjunto C formado por todas as classes, incluindo o conjunto vazio. Esse subconjunto pode ser representado por um vetor $C' = (c'_1, \dots, c'_{|C|})$, em que a classe c'_j pertence ao subconjunto de classes do exemplo $\mathbf{x}_i$ se e somente se $c'_j = 1$. Antes de definir a função **H-Loss**, duas medidas que medem a discrepância entre uma predição multirrótulo para um exemplo ($C = (c_1, \dots, c_{|C|})$), e o conjunto verdadeiro de classes de um exemplo ($C' = (c'_1, \dots, c'_{|C|})$) precisam ser introduzidas. A primeira delas é a medida *zero-one loss* ($l_{0/1}(C, C')$), apresentada na Equação

3.6, e a segunda delas é chamada *symmetric difference loss* ($l_\Delta(C, C')$), definida na Equação 3.7. Essas equações não consideram a estrutura hierárquica do problema, mas apenas as múltiplas classes. Baseado nessas duas medidas, Cesa-Bianchi et al. (2006) propuseram a medida *H-Loss* ($l_H(C, C')$), definida na Equação 3.8. Nas Equações, $\mathbb{1}\{\cdot\}$ é uma função que retorna 1 se a equação fornecida como argumento for verdadeira e 0 caso contrário.

$$l_{0/1}(C, C') = 1, \text{ if } \exists j \in \{1, \dots, |C|\} : c_j \neq c'_j \quad (3.6)$$

$$l_\Delta(C, C') = \sum_{j=1}^{|C|} \mathbb{1}\{c_j \neq c'_j\} \quad (3.7)$$

$$l_H(C, C') = \sum_{j=1}^{|C|} \mathbb{1}\{c_j \neq c'_j \wedge \text{Ancestrais}(c_j) = \text{Ancestrais}(c'_j)\} \quad (3.8)$$

A medida *H-Loss* é baseada no fato de que, dada uma estrutura hierárquica H , essa estrutura pode ser considerada uma floresta composta por árvores definidas sobre o conjunto de classes do problema. Uma classificação multirrótulo $C' \in \{0, 1\}^{|C|}$ respeita a estrutura H se e somente se C' for a união de um ou mais caminhos de H , sendo que cada caminho começa na raiz mas não necessariamente termina em uma classe folha. Assim, todos os caminhos de H , da raiz até uma folha, são examinados. Sempre que uma classe c_j for encontrada tal que $c_j \neq c'_j$, o valor 1 é adicionado ao valor da função *H-Loss*, e todas as predições nas subárvores enraizadas na classe c_j são descartadas. Dada essa definição, pode ser dito que $l_{0/1} \leq l_H \leq l_\Delta$.

A Figura 3.5 ilustra os conceitos e o uso da medida *H-Loss*. Nas quatro hierarquias de classes ilustradas, os nós representados pelos círculos cinza representam as classes preditas de um exemplo, enquanto os nós representados pelos quadrados cinza representam as classes reais de um exemplo. Pode-se notar na Figura 3.5(a) que as classes preditas não respeitam a estrutura hierárquica de H (os exemplos recebem classes conflitantes nos níveis hierárquicos 3 e 4), enquanto na Figura 3.5(b) a estrutura hierárquica é respeitada. A Figura 3.5(c) mostra as classes verdadeiras do exemplo classificado na Figura 3.5(b), e a Figura 3.5(d) mostra a aplicação da função *H-Loss* considerando as classificações multirrótulo ilustradas em (b) e (c). Somente os nós marcados com um “X” são considerados no cálculo da função *H-Loss*. Na figura, os valores das medidas *zero-one loss* e *symmetric difference loss* são 1 e 6 respectivamente. O valor da função *H-Loss* é igual a 4. Deve ser observado que quanto menor o valor da função *H-Loss*, melhor é o desempenho do classificador.

Figura 3.5: Representação gráfica da medida **H-Loss**. (Cesa-Bianchi et al., 2006)

Outras Medidas de Avaliação Baseadas em Hierarquia

Outra medida de avaliação baseada em hierarquia foi proposta por Ipeirotis et al. (2001). Como em (Kiritchenko et al., 2005), os autores mediram a **PH** e a **RH** por meio da intersecção das classes preditas e verdadeiras de um exemplo. A diferença dessa medida é que, diferente da medida proposta por Kiritchenko et al. (2005), ela expande o conjunto de classes incluindo todas as suas subclasses ao invés de suas superclasses. Assim, dados os conjuntos de classes preditas (C_i) e verdadeiras (C'_i), esses conjuntos são expandidos para conter suas correspondentes classes descendentes da seguinte maneira: $\widehat{C}_i = \bigcup_{c_k \in C_i} \text{Descendentes}(c_k)$ e $\widehat{C}'_i = \bigcup_{c_l \in C'_i} \text{Descendentes}(c_l)$, em que $\text{Descendentes}(c_k)$ denota o conjunto de descendentes da classe c_k . Essas novas definições de $\widehat{C}_i$ e $\widehat{C}'_i$ podem ser diretamente utilizadas na Equações 3.3 e 3.4. Apesar de os autores afirmarem que essa medida captura as nuances da classificação hierárquica, ela parece não ser totalmente adequada para problemas de classificação hierárquica multirrótulo, porque expandir um conjunto de classes para conter suas respectivas subclasses pode resultar em uma classificação errada. Como exemplo, se um documento é classificado na classe “esportes”, ele não é necessariamente classificado em ambas as subclasses “basquete” e “futebol”.

No trabalho de Sangsuriyun et al. (2010), foi proposta uma medida que combina a utilização da medida-F e do coeficiente de Jaccard ponderado para verificar a similaridade entre conjuntos de classes preditas e verdadeiras. O coeficiente de Jaccard ponderado entre um conjunto de ancestrais de classes preditas ($\widehat{C}_i$) e um conjunto de ancestrais classes verdadeiras ($\widehat{C}'_i$) é dado pela Equação 3.9.

$$Jaccard_w(\widehat{C}_i, \widehat{C}'_i) = \frac{\sum_{\forall j \in (\widehat{C}_i \cap \widehat{C}'_i)} w_j}{\sum_{\forall k \in (\widehat{C}_i \cup \widehat{C}'_i)} w_k} \quad (3.9)$$

Para o cálculo da medida de avaliação proposta ($Sim(C_i, C'_i)$), são considerados, no cálculo da medida-F, os conjuntos de classes preditas (C_i) e verdadeiras (C'_i) sem suas respectivas classes ancestrais. A medida de avaliação proposta é dada pela Equação 3.10.

$$Sim(C_i, C'_i) = \alpha \times \text{medida-F}(C_i, C'_i) + (1 - \alpha) \times Jaccard_w(\widehat{C}_i, \widehat{C}'_i) \quad (3.10)$$

Os pesos dos nós da hierarquia envolvidos no cálculo do coeficiente de Jaccard ponderado foram calculados de acordo com o conteúdo da informação que os nós representam. O conteúdo da informação de um nó c_j , $CI(c_j)$, pode ser definido por $CI(c_j) = -\log[P(c_j)]$, que em $P(c_j)$ é a probabilidade de ocorrência do nó c_j ou de ocorrência de algum de seus ancestrais mais específicos (Blanchard et al., 2005).

Em (Vens et al., 2008), uma medida de avaliação baseada na análise de curvas Precisão-Revocação (PR) foi utilizada. De acordo com (Davis e Goadrich, 2006), curvas PR são mais informativas a respeito do desempenho de um algoritmo quando utilizado em conjuntos de dados desbalanceados. Uma curva PR reflete a precisão de um classificador como uma função de sua revocação. Considere um classificador cl que tem como saída a probabilidade de um exemplo $\mathbf{x}_i$ ser atribuído a uma classe c_j , ou seja, sua saída é um valor real entre 0 e 1, e que a pertinência do exemplo $\mathbf{x}_i$ à classe c_j seja determinada de acordo com um dado limiar t . Assim, para valores de saída acima do valor t , o exemplo $\mathbf{x}_i$ é atribuído à classe c_j , e para valores abaixo de t , o exemplo $\mathbf{x}_i$ não é atribuído à classe c_j . De acordo com os autores, o uso de curvas PR é mais adequado para os problemas de classificação hierárquica multirrótulo porque, nesses problemas, classes individuais geralmente possuem poucos exemplos positivos.

Para comparar o desempenho de diferentes classificadores, os autores variaram o valor de limiar t , obtendo uma curva PR média. Duas abordagens foram utilizadas: a área abaixo da curva PR média e a área média ponderada abaixo das curvas PR individuais (por classe). A área abaixo da curva PR média é denotada por $AU(\overline{PRC})$. Quanto mais perto de 1 for o valor de $AU(\overline{PRC})$, melhor o desempenho do classificador. A média ponderada das áreas abaixo das curvas PR individuais é obtida de acordo com a Equação 3.11.

$$\overline{AUPRC}_{w_1, \dots, w_{|C|}} = \sum_j w_j \times AU(\overline{PRC})_j \quad (3.11)$$

Para o cálculo da média ponderada, os autores utilizaram dois tipos de inicialização para os pesos. O primeiro inicializa todos os pesos com o valor $1/|C|$, em que C é o conjunto de classes. Nesse caso, a medida foi denotada $\overline{AUPRC}$. Outra alternativa foi inicializar os pesos de acordo com as frequências das classes, ou seja, $w_j = v_j / \sum_k v_k$, com v_j a frequência da classe c_j no conjunto de dados. Essa instanciação foi utilizada porque em algumas aplicações classes mais frequentes podem ser mais importantes (Vens et al., 2008). Nesse caso, os autores denotaram a medida como $\overline{AUPRC}_w$.

De acordo com os autores, a vantagem da utilização de curvas PR na avaliação é a possibilidade de analisar muitos valores de limiar, e portanto observar melhor a relação entre os valores de precisão e revocação. A Figura 3.6 ilustra exemplos de curvas PR geradas utilizando o método descrito em (Vens et al., 2008).

Figura 3.6: Exemplos de curvas Precisão-Revocação.

As medidas de precisão e revocação utilizadas para a construção das curvas-PR podem ser as medidas de precisão e revocação hierárquicas definidas em (Kiritchenko et al., 2005). Em (Vens et al., 2008), os autores utilizaram as medidas apresentadas nas Equações 3.12 e 3.13. Nelas, i varia de 1 até o número total de classes $|C|$ do problema. Essas equações contam o número total de Verdadeiros Positivos (VP), Falsos Positivos (FP) e Falsos Negativos (FN) preditos. As equações podem ser utilizadas exatamente como as equações de precisão e revocação hierárquicas definidas em (Kiritchenko et al., 2005). Basta que as predições dos classificadores respeitem as restrições impostas pelas hierarquias.

$$\overline{Prec} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FP_i} \quad (3.12)$$

$$\overline{Rec} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FN_i} \quad (3.13)$$

3.5.2 Medidas de Avaliação Baseadas em Distância

Essa classe de medidas é baseada na ideia de que classes mais próximas na hierarquia tendem a ser mais similares entre si do que classes mais distantes. Assim, elas consideram as distâncias entre as classes preditas e verdadeiras no momento da avaliação da classificação. Esta seção apresenta duas medidas que possuem essas características, chamadas Micro/Macro Precisão Hierárquica (PH) baseadas em distância e Micro/Macro Revocação Hierárquica (RH) baseadas em distância. Adicionalmente, outras medidas de avaliação baseadas em distância são apresentadas.

Micro/Macro Precisão Hierárquica Baseada em Distância e Micro/Macro Revocação Hierárquica Baseada em Distância

As medidas Micro/Macro PH e Micro/Macro RH baseadas em distâncias entre classes foram propostas em (Sun e Lim, 2001; Sun et al., 2003). As medidas Macro Precisão e Revocação Hierárquicas medem o desempenho obtido em cada classe separadamente, e então calculam a média desses desempenhos. As medidas Micro Precisão e Revocação Hierárquicas por outro lado, medem o desempenho obtido em cada exemplo do conjunto de dados. Assim, enquanto as macro medidas são consideradas medidas de avaliação média por classe, as micro medidas são consideradas medidas de avaliação média por exemplo (Yang, 1999).

Em cada uma dessas medidas é calculada, para cada classe, a contribuição dos exemplos erroneamente atribuídos àquela classe. Essa contribuição é calculada de acordo com uma distância aceitável (Dis_θ) entre duas classes, que deve ter valor maior que zero. As Equações 3.14 e 3.15 apresentam os cálculos das contribuições de um exemplo $\mathbf{x}_i$ a uma classe c_j , em que $\mathbf{x}_i.agd$ e $\mathbf{x}_i.lbd$ são, respectivamente, as classes preditas e verdadeiras de $\mathbf{x}_i$.

- Se $\mathbf{x}_i$ for um Falso Positivo:

$$Con(\mathbf{x}_i, c_j) = \sum_{c' \in \mathbf{x}_i.lbd} \left(1.0 - \frac{Dis(c', c_j)}{Dis_\theta} \right) \quad (3.14)$$

- Se $\mathbf{x}_i$ for um Falso Negativo:

$$Con(\mathbf{x}_i, c_j) = \sum_{c' \in \mathbf{x}_i.agd} \left(1.0 - \frac{Dis(c', c_j)}{Dis_\theta} \right) \quad (3.15)$$

A contribuição de um exemplo $\mathbf{x}_i$ é então restrita ao valores $[-1, 1]$. Esse refinamento, denotado por $RCon(\mathbf{x}_i, c_j)$, é definido na Equação 3.16.

$$RCon(\mathbf{x}_i, c_j) = \min(1, \max(-1, Con(\mathbf{x}_i, c_j))) \quad (3.16)$$

As contribuições totais de **FP** ($FpCon_j$) e **FN** ($FnCon_j$), para todos os exemplos, são então definidas nas Equações 3.17 and 3.18.

$$FpCon_j = \sum_{\mathbf{x}_i \in FP_j} RCon(\mathbf{x}_i, c_j) \quad (3.17)$$

$$FnCon_j = \sum_{\mathbf{x}_i \in FN_j} RCon(\mathbf{x}_i, c_j) \quad (3.18)$$

Após o cálculo das contribuições de cada exemplo, os valores de PH^{CD} e RH^{CD} baseados em distância para cada classe são calculados como definido nas Equações 3.19 e 3.20.

$$PH_j^{CD} = \frac{\max(0, |VP_j| + FpCon_j + FnCon_j)}{|VP_j| + |FP_j| + FnCon_j} \quad (3.19)$$

$$RH_j^{CD} = \frac{\max(0, |VP_j| + FpCon_j + FnCon_j)}{|VP_j| + |FN_j| + FpCon_j} \quad (3.20)$$

Finalmente, os valores estendidos de **PH** e **RH** (Micro Precisão e Revocação Hierárquicas) são calculados utilizando as Equações 3.21 e 3.22, em que m representa o número de classes. Dependendo do valor de Dis_θ , os valores de $FpCon_j$ e $FnCon_j$ podem ser negativos. Assim, uma função $\max$ é aplicada aos numeradores das Equações 3.21 e 3.22 para garantir que seus valores sejam maiores que zero. Como $FpCon_j \leq |FP_j|$, quando $|VP_j| + |FP_j| + FnCon_j \leq 0$, o numerador $\max(0, |VP_j| + FpCon_j + FnCon_j) = 0$. O valor $\hat{PH}^{\mu CD}$ pode ser considerado zero nesse caso. A mesma regra é aplicada no cálculo de $\hat{RH}^{\mu CD}$ (Sun e Lim, 2001).

$$\hat{P}H^{\mu CD} = \frac{\sum_{j=1}^m (\max(0, |VP_j| + FpCon_j + FnCon_j))}{\sum_{j=1}^m (|VP_j| + |FP_j| + FnCon_j)} \quad (3.21)$$

$$\hat{R}H^{\mu CD} = \frac{\sum_{j=1}^m (\max(0, |VP_j| + FpCon_j + FnCon_j))}{\sum_{j=1}^m (|VP_j| + |FN_j| + FpCon_j)} \quad (3.22)$$

Os valores das medidas Macro PH e Macro RH também podem ser obtidos utilizando as Equações 3.23 e 3.24, em que m representa o número de classes.

$$\hat{P}H^{MCD} = \frac{\sum_{j=1}^m PH_j^{CD}}{m} \quad (3.23)$$

$$\hat{R}H^{MCD} = \frac{\sum_{j=1}^m RH_j^{CD}}{m} \quad (3.24)$$

Assim como nas medidas PH e RH, utilizadas em (Kiritchenko et al., 2005), as medidas Micro/Macro Precisão e Revocação Hierárquicas também podem ser combinadas na medida *Hierarchical- F_β* (Equação 3.5).

Outras Medidas de Avaliação Baseadas em Distância

Uma medida hierárquica probabilística, chamada Acurácia Hierárquica Probabilística (*Probabilistic Hierarchical Accuracy* (PHA)), foi proposta por Itskevitch (2001). Supondo que um classificador atribua a um exemplo $\mathbf{x}_i$ a seguinte distribuição de probabilidade: $(c_1 : p_1), (c_2 : p_2), \dots, (c_n : p_n)$, com p_j , $j = 1, \dots, m$, a probabilidade atribuída à classe c_j . A acurácia do classificador na classificação do exemplo $\mathbf{x}_i$ é então dada pela Equação 3.25.

$$PHA(\mathbf{x}_i) = \sum_{j=1}^m A_j(\mathbf{x}_i) \quad (3.25)$$

Na Equação 3.25, $A_j(\mathbf{x}_i)$ é a acurácia do classificador na classe c_j , e m é o número total de classes. $A_j(\mathbf{x}_i)$ é calculada com base nas probabilidades de predição e nas distâncias (número de arestas) entre as classes reais e preditas:

$$A_j(\mathbf{x}_i) = \begin{cases} -\frac{p_j}{d_j} & \text{for } d_j = 1, 2, \dots, d_c \\ p_j \log_2 d_j & \text{for } d_j = d_c + 1, \dots, d_{max} \end{cases}$$

em que $d_j = d_j(c_k, c_j)$, $j = 1, \dots, m$, é a distância entre as classes c_k e c_j na hierarquia. Na fórmula, d_{max} é a distância máxima entre duas classes na hierarquia, e d_c é a máxima distância entre as classes preditas e verdadeiras que é satisfatória para os propósitos de classificação (Itskevitch, 2001). A acurácia final de classificação é então a média de $PHA(\mathbf{x}_i)$ sobre todos os $|T|$ exemplos, dada pela Equação 3.26.

$$PHA = \frac{\sum_{i=1}^{|T|} PHA(\mathbf{x}_i)}{|T|} \quad (3.26)$$

3.5.3 Medidas de Avaliação Baseadas em Semântica

Medidas de avaliação baseadas em semântica utilizam conhecimento semântico dos problemas e como esse conhecimento influencia a hierarquia de classes para definir como penalizar erros de classificação. Intuitivamente, se um classificador A erroneamente atribui um exemplo a uma dada classe c_A , mas essa classe é similar às classes verdadeiras do exemplo, esse erro de classificação deve ser menos penalizado do que outro erro de classificação cometido por um classificador B , que atribui o mesmo exemplo a uma classe c_B não similar às classes reais do exemplo. Esta seção revisa algumas medidas baseadas em semântica propostas na literatura.

Micro/Macro Precisão Hierárquica Baseada em Semântica e Micro/Macro Revocação Hierárquica Baseada em Semântica

O cálculo da similaridade baseada em semântica pode não ser obtido de maneira trivial. Nos trabalhos de [Sun e Lim \(2001\)](#) e [Sun et al. \(2003\)](#) foi proposto um cálculo de similaridade entre duas classes c_j e c_k , denotado por $CS(c_j, c_k)$, utilizando a distância cosseno entre os dois vetores de características das classes. O vetor de características de uma classe pode ser obtido por meio da soma dos vetores de características de todos os exemplos atribuídos àquela classe. Os vetores de características são definidos pelos conjuntos de atributos dos exemplos. Assim, sendo t_i a soma da característica i dos exemplos, em que $i \in \{1, \dots, N\}$ com N sendo o número total de características, e sendo w_i e v_i pesos atribuídos aos índices, os vetores de características de duas classes c_j e c_k podem ser representados como:

- $\mathbf{FV}_{c_j} = \{w_1 t_1, w_2 t_2, \dots, w_N t_N\};$
- $\mathbf{FV}_{c_k} = \{v_1 t_1, v_2 t_2, \dots, v_N t_N\}.$

A similaridade entre quaisquer duas classes pode ser calculada utilizando a Equação 3.27. A média de todas as similaridades par a par entre as m classes, denotada por ACS , é calculada utilizando a Equação 3.28

$$CS(c_j, c_k) = \frac{\sum_{n=1}^N (w_n \times v_n)}{\sqrt{\sum_{n=1}^N w_n^2 \times \sum_{n=1}^N v_n^2}} \quad (3.27)$$

$$ACS = \frac{2 \times \sum_{j=1}^m \sum_{k=j+1}^m CS(c_j, c_k)}{m \times (m - 1)} \quad (3.28)$$

Baseado na similaridade entre as classes, é possível verificar o quão similares são as classes verdadeiras ($\mathbf{x}_i.lbd$) e preditas ($\mathbf{x}_i.agd$) de um exemplo $\mathbf{x}_i$, e então estimar o quanto um exemplo x_i pode contribuir para uma classe c_j no cálculo da Precisão e Revocação. Essas contribuições são calculadas utilizando as Equações 3.29 e 3.30.

- Se $\mathbf{x}_i$ for um Falso Positivo

$$Con(\mathbf{x}_i, c_j) = \frac{\sum_{c' \in \mathbf{x}_i.lbd} (CS(c', c_j) - ACS)}{1 - ACS} \quad (3.29)$$

- Se $\mathbf{x}_i$ for um Falso Negativo:

$$Con(\mathbf{x}_i, c_j) = \frac{\sum_{c' \in \mathbf{x}_i.agd} (CS(c', c_j) - ACS)}{1 - ACS} \quad (3.30)$$

Assim como nas medidas baseadas em distância, a contribuição de um exemplo pode ser positiva ou negativa, dependendo de quão similares são suas classes preditas e verdadeiras em comparação com a similaridade média das classes (ACS). Assim, é necessário um refinamento dos valores das contribuições dos exemplos (Equação 3.16). A contribuição de todos os exemplos que pertencem ao conjunto de Falsos Positivos e Falsos Negativos também é calculada (Equações 3.17 e 3.18).

Assim como na seção anterior, medidas estendidas de Precisão (PH_i^{CS}) e Revocação (RH_i^{CS}) também podem ser obtidas (Equações 3.19 e 3.20), assim como as medidas Micro Precisão Hierárquica ($\hat{P}H^{\mu CS}$) e Micro Revocação Hierárquica ($\hat{R}H^{\mu CS}$) (Equações 3.21 e 3.22), e as medidas Macro Precisão Hierárquica ($\hat{P}H^{MCS}$) e Macro Revocação Hierárquica ($\hat{R}H^{MCS}$) (Equações 3.23 e 3.24). Finalmente, as medidas Micro/Macro Precisão e Revocação Hierárquicas também podem ser combinadas na medida *Hierarchical- F_β* , calculada utilizando a Equação 3.5.

Outras Medidas de Avaliação Baseadas em Semântica

Em (Lord et al., 2003) e (Jin et al., 2008), os autores utilizaram anotações da **GO** para medir as similaridades semânticas entre as classes de um conjunto de dados, baseado no conteúdo da informação dos termos (Jiang e Conrath, 1997; Lin, 1998). Os autores utilizaram o conjunto de dados de proteínas *SWISS-PROT-Human proteins* (Bairoch e Apweiler, 2000). Na **GO**, um conjunto de termos rotula um conceito sobre um domínio. Assim, os autores contaram o número de vezes que cada conceito da **GO** aparecia no conjunto de dados. Um conceito aparece se um termo, ou qualquer um de seus descendentes aparece. Assim, a probabilidade $p(c)$ de cada nó da hierarquia é dada pelo número de vezes que um conceito aparece, dividido pelo número de vezes que cada termo aparece. Com isso, os autores garantem que a probabilidade de cada nó aumenta a medida que o grafo é percorrido em direção à raiz, e que a probabilidade do nó raiz aparecer seja igual a 1.

Uma vez calculadas as probabilidades, os autores calcularam a similaridade semântica entre dois termos baseado na informação dos pais compartilhados por ambos os termos, de acordo com a Equação 3.31, em que $S(c_1, c_2) = Ancestrais(c_1) \cap Ancestrais(c_2)$. Como na estrutura hierárquica da **GO** pode haver mais de um pai para cada classe, dois termos da **GO** podem compartilhar ancestrais de múltiplos caminhos. Assim, quando foi o caso, os autores utilizaram a probabilidade mínima $p(c)$. Essa probabilidade foi denominada probabilidade do ancestral mais profundo na hierarquia (*probability of the minimum subsumer* (p_{ms})).

$$p_{ms}(c_1, c_2) = \min_{c \in S(c_1, c_2)} \{p(c)\} \quad (3.31)$$

A pontuação da similaridade semântica entre dois termos c_1 e c_2 da **GO** é então calculada utilizando a Equação 3.32. Essa nova medida não foi utilizada na avaliação de classificadores,

consistindo apenas de uma proposta teórica para o cálculo da similaridade entre classes.

$$\text{sim}(c_1, c_2) = -\ln p_{ms}(c_1, c_2) \quad (3.32)$$

Uma medida baseada em semântica para calcular a similaridade entre classes também foi descrita por Wang e Senqiang (2001), em que a similaridade de dois conjuntos de classes foi definida como a similaridade do conjunto de exemplos (documentos no caso) que pertencem às classes. Seja C'_i o conjunto de classes verdadeiras de um exemplo, e C_i o conjunto de classes preditas de um exemplo. Assim, a cobertura de C_i , denotada por $\text{Cover}(C_i)$, é o conjunto de todos os exemplos cobertos por C_i . O erro de classificação entre dois conjuntos de classes ($\text{ClErr}(C'_i, C_i)$) é então dado pela Equação 3.33.

$$\text{ClErr}(C'_i, C_i) = \frac{|\text{Cover}(C_i) - \text{Cover}(C'_i)| + |\text{Cover}(C'_i) - \text{Cover}(C_i)|}{|\text{Cover}(C'_i) \cup \text{Cover}(C_i)|} \quad (3.33)$$

3.5.4 Contrastando Diferentes Medidas de Avaliação

Ambas as medidas baseadas em distância e baseadas em hierarquia possuem a desvantagem de não considerarem que as predições feitas nos níveis mais profundos da hierarquia são mais difíceis e levam a informações mais específicas do que predições realizadas nos níveis mais próximos à raiz. Essa desvantagem pode ser contornada fazendo com que os custos dos erros de classificação nas classes mais próximas à raiz sejam maiores do que nas classes mais próximas às folhas.

Considerando a distância entre duas classes, essa pode ser definida como uma função de dois componentes: o número de arestas entre a classe predita e a classe verdadeira, e a profundidade das classes preditas e verdadeiras na hierarquia. Uma maneira simples de se calcular essa função é por meio da atribuição de pesos a cada aresta da hierarquia. Dessa forma, o erro de classificação entre uma classe predita e uma classe verdadeira é dado pela soma dos pesos das arestas no caminho entre as duas classes. Uma maneira de escolher os pesos que serão atribuídos a cada aresta é considerar que os pesos das arestas localizadas nos níveis mais profundos devem ser menores do que os pesos das arestas localizadas nos níveis mais elevados. No trabalho de Holden e Freitas (2006), por exemplo, os pesos atribuídos às arestas foram decrementados a medida que a profundidade das arestas aumentava.

Existem, entretanto, alguns problemas na atribuição de pesos às arestas da hierarquia, especialmente quando a hierarquia for desbalanceada, e sua profundidade varia significativamente em diferentes nós folhas. Nesse caso, um erro de classificação envolvendo classes verdadeiras e preditas perto do nó raiz recebe uma penalização menor do que um erro de classificação envolvendo classes localizadas em níveis mais distantes do nó raiz.

O problema da atribuição de pesos às arestas da hierarquia também foi considerado por Lord et al. (2003). De acordo com os autores, quando duas classes estão localizadas em diferentes subárvores da hierarquia, e o caminho entre essas classes tem que passar pelo nó raiz, o fato de uma classe estar localizada em um nível mais profundo do que outra não significa necessaria-

mente que a classe localizada no nível mais profundo representa informação mais específica do que a classe localizada em um nível mais elevado. Portanto, considerar apenas a profundidade sem considerar as informações associadas às classes, pode ser um problema em alguns casos.

A Figura 3.7 ilustra o problema da atribuição de pesos às arestas da hierarquia. Na figura, as elipses sólidas representam as classes reais de um exemplo, e as elipses em negrito representam as classes preditas de um exemplo. A figura mostra um exemplo que pertence à classe “11.04.03.01” (classe real), e duas predições para esse exemplo, sendo que uma predição atribui a classe “11.02.03.04” ao exemplo (Predição 1) e outra predição atribui a classe “11.06.01” ao exemplo (Predição 2). No último caso, a predição recebe uma penalização menor, porque o caminho entre as classes predita e real é mais curto. Nesse caso, essa penalização foi injusta, porque a predição foi feita em uma classe mais perto da raiz devido ao fato da subárvore correspondente não possuir classes folhas no último nível.

Figura 3.7: Exemplo de hierarquia de classes.

Outro problema com as medidas baseadas em distância, e nesse caso também com as medidas baseadas em semântica, é que elas dependem de informações subjetivas e parâmetros informados pelos usuários, como nas medidas Micro/Macro Precisão e Revocação Hierárquicas baseadas em semântica e distância. A primeira medida depende de pesos atribuídos aos vetores de características das classes, enquanto a última medida depende de uma distância aceitável fornecida pelo usuário. A variação desses parâmetros pode mudar drasticamente os valores obtidos pelas medidas.

As distâncias entre as classes verdadeiras e preditas e a profundidade de uma classe são outros conceitos que dificultam a utilização de medidas baseadas em distância, tornando seu uso possível em apenas um tipo de estrutura hierárquica. Quando utilizando hierarquias estruturadas como árvores, cada classe pode ter apenas uma profundidade e, consequentemente, há apenas um possível caminho entre duas classes. Quando utilizando hierarquias estruturadas

como DAGs, a profundidade de uma classe não é mais única, pois uma classe pode possuir mais de um pai na hierarquia. Assim, pode haver mais de um caminho entre duas classes na hierarquia, dificultando a utilização das medidas.

Apesar de alguns autores utilizarem a medida **H-Loss**, seu uso também apresenta alguns problemas. Devido ao fato de essa medida ignorar erros de classificação cometidos nas subárvores das classes classificadas erroneamente, o valor de desempenho apresentado por essa medida parece injusto se comparado com as outras medidas de avaliação. Devido a essa característica, o problema de propagação de erros presente na classificação hierarquia não é levado em consideração. Ainda, experimentos realizados em (Cerri et al., 2013) sugerem que essa medida tende a favorecer métodos de classificação baseados na abordagem local, e portando não deve ser utilizada em experimentos comparando as abordagens local e global.

De maneira a exemplificar como a medida **H-Loss** pode favorecer métodos de classificação locais, serão consideradas as hierarquias ilustradas na Figura 3.8. Na figura, os círculos sólidos representam as classes que foram atribuídas a um exemplo. A Figura 3.8(a) representa as classes verdadeiras do exemplo, enquanto as Figuras 3.8(b) e 3.8(c) representam, respectivamente, as classes atribuídas ao exemplo por um método global e por um método local. Como pode ser observado na figura, mesmo que a medida **H-Loss** considere os erros cometidos em classes mais próximas da raiz mais importantes do que os erros cometidos em classes mais profundas, o cálculo da medida **H-Loss** resulta no valor 2 para o método local e no valor 4 para o método global. Nesse caso, pode ser visto como a medida favoreceu o método local, mesmo que a classificação fornecida pelo método local tenha sido pior que a classificação fornecida pelo método global. É claro que esse cenário pode ser facilmente invertido se for considerado que o método global fez as previsões representadas na Figura 3.8(c) e que o método local fez as previsões representadas na Figura 3.8(b). No entanto, nos experimentos realizados em (Cerri et al., 2013), notou-se que o cenário representado na Figura 3.8 aconteceu na maioria dos casos. Ainda, os métodos globais possuem a característica que prever mais classes localizadas nos níveis mais profundos, o que favorece a ocorrência do cenário apresentado na figura.

Figura 3.8: Exemplo de como a medida **H-Loss** pode favorecer a abordagem local.

Assim, devido aos problemas presentes em muitas medidas de avaliação, e também porque algumas medidas presentes na literatura ou são apenas utilizadas pelos seus autores, ou não estão claramente definidas, o trabalho de Silla e Freitas (2010) recomenda a utilização das medidas PH e RH propostas por Kiritchenko et al. (2005). De acordo com Silla e Freitas (2010), apesar de nenhuma medida de avaliação poder ser considerada a melhor em todos os possíveis cenários e aplicações envolvendo classificação hierárquica, a razão para essa recomendação é que essas medidas podem ser aplicadas diretamente em qualquer cenário envolvendo classificação hierárquica. Adicionalmente, elas não dependem de informações subjetivas e parâmetros especificados pelos usuários. Ainda, em (Cerri et al., 2013), as medidas PH e RH foram comparadas experimentalmente com outras medidas hierárquicas multirrótulo. Além de nenhuma desvantagem ser detectada na utilização das medidas de precisão e revocação hierárquicas, essas mostraram-se consistentes com outras medidas investigadas, possuindo as vantagens supracitadas.

As medidas PH e RH podem ser utilizadas para a construção de curvas Precisão-Revocação. Como já mencionado, curvas PR têm sido utilizadas em problemas de recuperação de informação (Manning e Schütze, 1999; Raghavan et al., 1989) e também, mais recentemente, em problemas de classificação hierárquica multirrótulo (Vens et al., 2008; Otero et al., 2010). A vantagem de se utilizar esse tipo de medida de avaliação é que a medida é mais adequada para conjuntos de dados desbalanceados (um grande número de exemplos negativos e um pequeno número de exemplos positivos), fazendo com que o número de verdadeiros negativos não influencie na obtenção dos valores de Precisão e Revocação. Isso é importante no momento de se avaliar uma classificação hierárquica multirrótulo porque, como mencionado anteriormente, os exemplos localizados em níveis mais profundos possuem poucos exemplos positivos. Nesses problemas, é mais importante medir o quão bem um classificador prediz a presença de uma classe particular (verdadeiro positivo) do que sua ausência (verdadeiro negativo).

3.6 Considerações Finais

Este capítulo apresentou os conceitos fundamentais de problemas de classificação hierárquica e multirrótulo, além das principais abordagens utilizadas para tratar esses tipos de problemas. Também foi feita uma revisão bibliográfica sobre vários trabalhos que tratam problemas hierárquicos multirrótulo, categorizando-os de acordo com a abordagem utilizada para resolver o problema e também de acordo com a estrutura hierárquica na qual cada método é capaz de ser aplicado. Uma revisão sobre medidas de avaliação também foi feita, separando as medidas em três grupos principais, e discutindo as vantagens e desvantagens de cada grupo. O próximos capítulos apresentam os métodos propostos nessa pesquisa de doutorado.

Classificação Hierárquica Multirrótulo com Perceptrons Multi-Camadas Locais

Este capítulo apresenta o método para classificação hierárquica multirrótulo utilizando perceptrons multi-camadas locais, do inglês *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* (**HMC-LMLP**). Trata-se de um método que tem como objetivo dividir o processo de aprendizado em sub-processos, de forma a induzir um modelo de classificação complexo por meio da combinação de modelos mais simples induzidos de maneira sequencial. O método segue a estratégia de classificação *Local Classifier per Level* (**LCL**), e a ideia é que cada modelo em sequência seja forçado a aprender algo diferente dos modelos previamente treinados, tentando assim evitar *overfitting*. Para uma melhor compreensão do funcionamento do método, será utilizado o conjunto de dados fictício apresentado na Figura 4.1. Na figura, as classes do conjunto de dados são estruturadas em uma hierarquia do tipo árvore com três níveis de profundidade.

u

Os modelos utilizados no método **HMC-LMLP** são redes neurais do tipo *Multi-Layer Perceptron* (**MLP**), treinadas sequencialmente, uma para cada nível da hierarquia de classes. Sendo assim, cada **MLP** é associada a um nível, e torna-se responsável pela predição das classes pertencentes ao seu nível associado. Devido a essa característica, que torna necessária a explícita especificação do nível em que encontra-se uma classe, o método foi desenvolvido especificamente para hierarquias estruturadas como árvores, já que em hierarquias estruturadas como **DAGs** uma classe pode estar localizada em diferentes níveis, dependendo do caminho percorrido entre a raiz e a classe. Entretanto, uma adaptação pode ser feita em hierarquias estruturadas como **DAGs**. Pode-se, por exemplo, definir o nível de uma classe como o número de arestas do caminho mais curto entre o nó raiz e a classe.

Quatro variações do método **HMC-LMLP** foram propostas. A primeira utiliza as predições

Figura 4.1: Exemplo de uma hierarquia de classes de três níveis. O símbolo “.” separa classes de superclasses/subclasses (02.11 significa que 02 é uma superclasse de 11), e o símbolo “/” separa diferentes caminhos na hierarquia atribuídos a um exemplo.

obtidas por uma rede **MLP** associada a um nível como as únicas entradas fornecidas à rede **MLP** associada ao próximo nível. A segunda utiliza as predições obtidas por uma rede **MLP** em um nível para complementar os vetores de atributos preditivos dos exemplos a serem fornecidos como entrada à rede **MLP** associada ao próximo nível. Em uma terceira variação, aos invés das predições, são utilizadas as classes verdadeiras dos exemplos para complementar seus vetores de atributos durante o treinamento das redes neurais. A quarta variação treina uma rede neural **MLP** para cada nível hierárquico utilizando apenas os atributos originais dos exemplos, ou seja, nenhuma informação com relação às classes preditas ou verdadeiras é utilizada.

As próximas seções apresentam detalhadamente as quatro variações do método **HMC-LMLP**. A variação que utiliza apenas as predições como entradas das redes **MLP** será chamada de **HMC-LMLP-Labels** e apresentada na Seção 4.1; a Seção 4.2 apresenta a variação que utiliza as predições para complementar os vetores de atributos dos exemplos, chamada de **HMC-LMLP-Predicted**; quando referindo-se à variação que utiliza as classes verdadeiras para complementar os vetores de atributos, será utilizado o termo **HMC-LMLP-True**. Essa variação é apresentada na Seção 4.3; já a variação que não utiliza nenhuma informação com relação às classes será chamada de **HMC-LMLP-NoLabels**, e apresentada na Seção 4.4; a maneira como as predições do método **HMC-LMLP** são obtidas é apresentada na Seção 4.5; na Seção 4.6 é

apresentada a complexidade computacional das quatro variações propostas; por fim, a Seção 4.7 apresenta as considerações finais do Capítulo.

4.1 Predições como Única Entrada - HMC-LMLP-Labels

A variação HMC-LMLP-Labels é ilustrada na Figura 4.2. Os valores fornecidos pelas funções de ativação da camada de saída da MLP associada ao nível l são usados como entrada para a MLP associada ao nível $l + 1$. Na figura, $\mathbf{X}$ representa os exemplos de treinamento; h_l e O_l representam, respectivamente, a camada escondida e a camada de saída da rede neural associada ao nível l . As matrizes $\mathbf{W}_{1l}$ e $\mathbf{W}_{2l}$ representam, respectivamente, os pesos conectando a camada de entrada à camada escondida, e os pesos conectando a camada escondida à camada de saída da rede neural associada ao nível l . O número de neurônios de saída de cada camada de saída das MLPs da Figura 4.2 é igual ao número de classes de seu nível associado na hierarquia ilustrada na Figura 4.1. Como pode ser observado, a rede neural associada ao nível 2 tem seis neurônios em sua camada de saída. O mesmo pode ser observado nas redes neurais associadas aos outros níveis hierárquicos. Como resultado, as MLPs podem prever mais de uma classe simultaneamente, pois cada classe de um nível hierárquico é associada a um neurônio de saída da rede neural. Os valores de saída preditos são então comparados com os valores de saída desejados para os exemplos daquele nível para que seja calculado o erro da rede no nível.

Figura 4.2: Variação HMC-LMLP-Labels. (a) treinamento de uma MLP para o primeiro nível hierárquico; (b) utilização da saída da primeira MLP para treinamento de uma MLP associada ao segundo nível; (c) utilização da saída da segunda MLP para treinamento de uma MLP associada ao terceiro nível.

O processo de treinamento é realizado da seguinte maneira: inicialmente, uma rede neural MLP é associada com o primeiro nível da hierarquia, tendo como entrada os exemplos de treinamento $\mathbf{X}$. Essa rede MLP pode ser treinada utilizando qualquer algoritmo de aprendizado para redes MLP, e sua arquitetura possui uma camada escondida. Depois que o processo de treinamento dessa MLP para o primeiro nível é finalizado, como ilustrado na Figura 4.2(a), uma segunda rede MLP é associada ao próximo nível da hierarquia. Essa rede também possui uma camada escondida, e seu treinamento ocorre da mesma maneira que a rede neural anterior. A única diferença é que a entrada dessa rede neural é agora a saída fornecida pela rede neural

associada ao nível anterior, e previamente treinada, como mostrado na Figura 4.2(b). Esse processo de treinamento incremental de uma rede neural para cada nível hierárquico é executado até que o último nível seja alcançado.

Durante o treinamento de uma **MLP** para um nível hierárquico específico, as **MLPs** associadas aos níveis anteriores não são treinadas novamente, pois seus treinamentos ocorreram nas etapas anteriores do treinamento do método **HMC-LMLP**. Quando uma rede **MLP** é treinada para um nível l , as $l - 1$ redes neurais treinadas anteriormente são utilizadas apenas para fornecer as entradas para o treinamento da rede neural associada ao nível l . Assim, durante o treinamento de uma rede neural para o nível l , suas entradas são obtidas utilizando os exemplos de treinamento na rede neural associada ao primeiro nível hierárquico. Os valores de saída dessa primeira **MLP** são então utilizados como entrada para a segunda **MLP** associada ao segundo nível da hierarquia. Essa segunda rede neural então fornece sua saída para ser utilizada como entrada para a próxima rede neural. Esse processo continua até que a rede **MLP** associada ao nível l seja alcançada. Para a classificação de novos exemplos, esses são fornecidos à rede neural associada ao primeiro nível. Os valores de saída dessa rede neural são então utilizados como entrada para a rede neural associada ao segundo nível. Esse processo é repetido nível a nível até que o último nível hierárquico seja alcançado.

4.2 *Predições como Complementação dos Vetores de Atributos - HMC-LMLP-Predicted*

Como pode ser observado na Figura 4.2, a primeira variação do método **HMC-LMLP** (**HMC-LMLP-Labels**) faz uso dos exemplos de treinamento como entrada apenas na rede neural associada ao primeiro nível da hierarquia. Do segundo nível em diante, apenas as predições feitas no nível anterior são utilizadas como entrada para as redes **MLP**. Essa estratégia possui a desvantagem de que as informações locais dos exemplos são utilizadas apenas no primeiro nível, ou seja, as informações fornecidas pelos atributos preditivos dos exemplos são utilizadas apenas no primeiro nível da hierarquia. Assim, deixam de ser utilizadas informações locais importantes (atributos) dos exemplos associados aos próximos níveis. Observou-se que isso levou a desempenhos de classificação ruins do segundo nível em diante.

Para contornar esse problema, foi proposta a variação **HMC-LMLP-Predicted**. Nessa variação, os exemplos de treinamento são utilizados no processo de treinamento de todas as redes **MLP** associadas a todos os níveis da hierarquia. Em outras palavras, durante o processo de treinamento, a entrada utilizada para a indução da rede **MLP** associada ao nível l agora inclui os exemplos de treinamento que são classificados nas classes pertencentes ao nível l . Dessa maneira, cada rede **MLP** agora extrai informações locais dos exemplos pertencentes aos seus níveis associados, o que acredita-se ser útil para a generalização e classificação de novos exemplos. Como problemas hierárquicos multirrótulo são complexos, trabalha-se com a hipótese de que diferentes informações podem ser extraídas dos exemplos de treinamento em diferentes níveis hierárquicos.

A Figura 4.3 ilustra a arquitetura e o processo de treinamento da variação HMC-LMLP-Predicted para a indução de um modelo de classificação para o conjunto de dados ilustrado na Figura 4.1. Na figura, $\mathbf{X}^l$ representa os exemplos de treinamento classificados nas classes localizadas no nível l da hierarquia. Cada rede MLP agora tem como entrada os exemplos de treinamento classificados nas classes de seu nível associado. Adicionalmente, os vetores de atributos dos exemplos utilizados no treinamento da rede MLP associada ao nível l são complementados com as saídas fornecidas pela rede MLP associada ao nível $l - 1$.

Figura 4.3: Variação HMC-LMLP-Predicted. (a) treinamento de uma MLP para o primeiro nível hierárquico; (b) utilização da saída da primeira MLP para complementar os vetores de atributos dos exemplos utilizados para o treinamento da MLP associada ao segundo nível; (c) utilização da saída da segunda MLP para complementar os vetores de atributos dos exemplos utilizados para o treinamento da MLP associada ao terceiro nível.

O processo de treinamento dessa nova variação do método HMC-LMLP é similar ao processo de treinamento da variação anterior. Primeiramente, uma rede neural é associada ao primeiro nível hierárquico, como ilustrado na Figura 4.3(a). Essa rede neural é treinada utilizando todos os exemplos de treinamento $\mathbf{X}^1$, já que todos os exemplos são classificados em classes do primeiro nível. Novamente, de maneira a permitir que uma rede neural prediga um conjunto de classes, cada neurônio de saída de uma MLP associada ao nível l é associado a uma classe pertencente ao nível l . Como a hierarquia ilustrada na Figura 4.1 possui duas classes no primeiro nível, a rede MLP associada ao segundo nível possui dois neurônios de saída. A rede neural possui uma camada escondida, e qualquer algoritmo de aprendizado para redes MLP pode ser utilizado em seu treinamento.

A rede neural associada ao segundo nível da hierarquia, como ilustrado na Figura 4.3(b), também possui uma camada escondida, e tem seu processo de treinamento executado de maneira similar à rede associada ao primeiro nível. A única diferença é que sua entrada é agora formada pelos exemplos de treinamento classificados em classes localizadas no segundo nível hierárquico ($\mathbf{X}^2$). Os vetores de atributos desses exemplos são também complementados com as saídas fornecidas pela rede neural treinada no nível anterior, ou seja, números reais utilizados

para classificar os exemplos no nível 1. Assim, as saídas da rede neural anterior complementam os vetores de características dos exemplos no segundo nível. A vantagem de se utilizar os vetores de características complementados para o treinamento das redes MLP é que a dependência entre as classes é incorporada no processo de aprendizado. Ideia similar foi proposta recentemente em (Cherman et al., 2012) e também em (Read et al., 2009) e (Dembczynski et al., 2010). Nesses trabalhos, as classes foram utilizadas para complementar os vetores de atributos dos exemplos de maneira a permitir que classificadores binários descobrissem sozinhos as dependências existentes entre as classes de problemas multirrótulo não hierárquicos.

O treinamento da rede neural associada ao terceiro nível hierárquico (Figura 4.3)(c) segue o mesmo procedimento utilizado para o treinamento da rede neural no segundo nível. No exemplo ilustrado na Figura 4.3, um modelo completo para uma hierarquia de três níveis foi construído. Esse procedimento de treinamento incremental pode ser utilizado em uma hierarquia de classes com um número de níveis qualquer. O processo de treinamento continua até que o último nível hierárquico seja alcançado.

Da mesma maneira que na variação HMC-LMLP-Labels, quando uma rede neural está sendo treinada para um nível específico, as redes MLP associadas aos níveis anteriores não são treinadas, pois seus respectivos treinamentos já ocorreram nas fases anteriores de indução (treinamento das MLPs associadas aos níveis anteriores). Quando uma rede MLP está sendo treinada para um nível l , a rede neural associada ao nível anterior é utilizada apenas para fornecer as saídas que serão utilizadas para complementar os vetores de atributos dos exemplos de treinamento no nível l . Assim, quando uma rede MLP é treinada para o nível l , suas entradas são obtidas fornecendo os exemplos de treinamento como entrada para a rede neural associada ao primeiro nível. A saída dessa rede neural é então utilizada para complementar os vetores de atributos dos exemplos utilizados no treinamento da rede neural associada ao segundo nível, que por sua vez fornece suas saídas para complementar os vetores de atributos dos exemplos usados no treinamento da próxima rede neural no próximo nível. Esse procedimento é executado até que a rede neural associada ao nível l seja alcançada.

Para a classificação de um novo exemplo, o mesmo é fornecido como entrada para a rede neural associada ao primeiro nível hierárquico, e a saída dessa rede MLP é então utilizada para complementar o vetor de atributos do exemplo. Esse vetor de atributos complementado com as saídas do nível anterior é então utilizado como entrada para a rede MLP associada ao segundo nível. Isso é feito para se obter as predições no segundo nível, que por sua vez são utilizadas para complementar o vetor de atributos do exemplo. Esse vetor de atributos, complementado com as predições no segundo nível, é fornecido como entrada para a rede MLP responsável pelas predições no terceiro nível. Esse processo é repetido até que a última rede MLP, associada ao último nível hierárquico, seja alcançada. É importante observar que o complemento dos vetores de atributos não é incremental, ou seja, o vetor de atributos de um exemplo sendo utilizado como entrada de uma rede MLP associada ao nível l é apenas complementado com as saídas fornecidas pela rede MLP associada ao nível $l - 1$.

De maneira a verificar se a proposta de utilização das predições para complementar os ve-

tores de atributos realmente resulta em melhora do desempenho de classificação, duas outras variações não incrementais do método [HMC-LMLP](#) foram implementadas. A primeira utiliza as classes verdadeiras dos exemplos para complementar seus vetores de atributos (HMC-LMLP-True), e a segunda não utiliza nenhuma informação de classes para complementar os vetores de atributos, ou seja, apenas os valores originais dos atributos dos exemplos são utilizados (HMC-LMLP-NoLabels). Essas duas variações são apresentadas nas próximas seções.

4.3 Classes Verdadeiras como Complementação dos Vetores de Atributos - HMC-LMLP-True

Da mesma maneira como é feito nas duas variações incrementais do método [HMC-LMLP](#) (HMC-LMLP-Labels e HMC-LMLP-Predicted), na variação HMC-LMLP-True uma rede [MLP](#) também é associada a cada nível hierárquico. No entanto, o treinamento das redes não acontece de maneira incremental, já que agora as redes não dependem mais umas das outras. Em cada nível hierárquico l , os vetores de atributos dos exemplos utilizados para treinar a rede neural associada ao nível l são complementados com as classes verdadeiras desses exemplos no nível $l - 1$. A Figura 4.4 ilustra esse processo de treinamento. Na figura, T_l representa as classes verdadeiras atribuídas a um exemplo no nível l .

Figura 4.4: Variação HMC-LMLP-True. (a) treinamento de uma [MLP](#) para o primeiro nível hierárquico; (b) utilização das classes verdadeiras do primeiro nível para complementar os vetores de atributos dos exemplos utilizados para o treinamento da [MLP](#) associada ao segundo nível; (c) utilização das classes verdadeiras do segundo nível para complementar os vetores de atributos dos exemplos utilizados para o treinamento da [MLP](#) associada ao terceiro nível.

Para a classificação de um novo exemplo, como o mesmo não possui nenhuma informação de suas classes, o procedimento adotado é o mesmo da variação HMC-LMLP-Predicted. Isso significa que as classes previstas para um exemplo no nível l são utilizadas para complementar o vetor de atributos do exemplo no próximo nível.

4.4 Vetores de Atributos como Única Entrada - HMC-LMLP-NoLabels

A Figura 4.5 ilustra a variação HMC-LMLP-NoLabels, que não utiliza nenhuma informação das classes no treinamento e teste das redes MLP. Como pode ser observado, as três redes neurais associadas a cada nível são treinadas separadamente, sendo que uma rede não depende da outra. Para a classificação de um novo exemplo, o mesmo é dado como entrada para todas as redes neurais separadamente, que então produzem predições independentes para cada nível.

Figura 4.5: Variação HMC-LMLP-NoLabels. (a) treinamento de uma MLP para o primeiro nível hierárquico; (b) treinamento de uma MLP para o segundo nível hierárquico; (c) treinamento de uma MLP para o terceiro nível hierárquico.

4.5 Obtendo as Predições Finais

Nas duas variações incrementais do método HMC-LMLP (HMC-LMLP-Labels e HMC-LMLP-Predicted) e também nas suas duas variações não incrementais (HMC-LMLP-True e HMC-LMLP-NoLabels), as predições finais são obtidas por meio da aplicação de valores de limiares nos neurônios de saída das redes MLP em cada nível. Se a saída de um dado neurônio n_j for igual ou superior a um dado valor de limiar, o exemplo sendo classificado é atribuído à classe c_j . A classificação final de um dado exemplo é então dada por um vetor binário $\mathbf{v}$ de tamanho $|C|$, com C o conjunto de todas as classes da hierarquia. Se o valor de saída do neurônio n_j associado a classe c_j em um dado nível for igual ou superior a um dado limiar, o valor 1 é atribuído à posição v_j do vetor, e 0 caso contrário. É esperado que a utilização de diferentes valores de limiares resulte em diferentes classes sendo preditas. Como a função de ativação utilizada nos neurônios de saída é a sigmoideal logística, os valores de saída de cada neurônio restringem-se ao intervalo $[0, 1]$. Dessa forma, pode-se utilizar valores de limiares também variando no intervalo $[0, 1]$. Quanto maior o valor de limiar utilizado, menor é o número de classes preditas, assim como quanto menor o valor de limiar utilizado, maior o número de classes preditas. É importante ressaltar que nas variações HMC-LMLP-Labels e HMC-LMLP-Predicted, os valores passados de uma rede para a outra são os valores obtidos das saídas das

redes neurais antes da aplicação dos limiares, ou seja, valores no intervalo $[0, 1]$. A aplicação dos limiares é utilizada somente para a obtenção da classificação final de um exemplo.

Após a obtenção das predições finais, uma fase de pós-processamento é necessária para correção de inconsistências que podem ocorrer na classificação final. Essas inconsistências ocorrem quando uma classe é predita sendo que sua superclasse imediata não é predita (no caso de hierarquias estruturadas como árvores), ou então quando uma classe é predita e uma ou mais de suas superclasses imediatas (no caso de hierarquias estruturadas como DAGs) não são preditas. Essas predições ocorrem porque cada rede MLP faz predições individuais em cada nível. Portanto, quando uma classe é predita em um nível l , não há garantias de que todas as suas superclasses nos outros $l - 1$ níveis anteriores foram preditas. Esse é um problema intrínseco da estratégia que utiliza um classificador local por nível (LCL) (Silla e Freitas, 2010).

Dois procedimentos diferentes são utilizados para a correção de inconsistências em hierarquias estruturadas como árvores e DAGs. Em árvores, a fase de pós-processamento simplesmente remove das classes preditas aquelas classes cujas superclasses não foram preditas. Já em DAGs, primeiramente é verificado se algum caminho completo foi predito entre uma dada classe e o nó raiz. Se de todos os possíveis caminhos entre a classe e a raiz, pelo menos um deles tiver sido totalmente predito, então a classe, e todas as suas superclasses em todos os outros caminhos, são consideradas na classificação final do exemplo. Caso contrário, a classe em questão e todas as outras classes pertencentes aos caminhos inconsistentes entre a classe e a raiz, que não tiverem superclasses preditas, são removidas da classificação final. A Figura 4.6 ilustra o processo de correção de inconsistências aplicado ao vetor final de predições provenientes de uma hierarquia estruturada como árvore.

Figura 4.6: Pós-processamento para correção de inconsistências. (a) saídas dos neurônios; (b) predição após aplicação do valor de limiar 0.5; (c) classificação final após correção de inconsistências.

Como pode ser observado na Figura 4.6(a), inicialmente tem-se um vetor de predições obtido por meio de quatro redes MLP associadas a cada nível de uma hierarquia de quatro níveis. A Figura 4.6(b) ilustra o estado do vetor após a aplicação de um limiar de valor 0.5. Todas as posições que possuíam valores iguais ou superiores a 0.5 receberam o valor 1, e 0 caso contrário. A Figura 4.6(c) apresenta a classificação final após o pós-processamento aplicado para

eliminar inconsistências. Os valores destacados representam as posições nas quais o valor 1 foi trocado pelo valor 0. A classificação final, consistente com a hierarquia de classes, atribui ao exemplo dois caminhos da hierarquia: 1.1.1.1 e 2.1.1.

4.6 Complexidade Computacional

Considerando a complexidade computacional do método [HMC-LMLP](#), as variações HMC-LMLP-Predicted e HMC-LMLP-True possuem complexidade igual a $O(W_l)$, sendo W_l o número de pesos e biases de cada rede [MLP](#) associada com o nível l . Considerando que $|A|$ é o número de atributos de um conjunto de dados, H_l é o número de neurônios da camada escondida da rede [MLP](#) associada ao nível l , e O_l é o número de neurônios de saída da [MLP](#) associada ao nível l , o número de pesos e biases no primeiro nível hierárquico (W_1) é definido como $(|A| + 1) \times H_1 + (H_1 + 1) \times O_1$. A partir do segundo nível, W_l é definido como $(O_{l-1} + |A| + 1) \times H_l + (H_l + 1) \times O_l$.

Na variação HMC-LMLP-NoLabels, o custo computacional é menor, já que as classes não são utilizadas para complementar os vetores de atributos dos exemplos. Assim, em todos os níveis l , a complexidade computacional é dada por $O(W_l)$, com $W_l = (|A| + 1) \times H_l + (H_l + 1) \times O_l$.

A variação HMC-LMLP-Labels tem a menor complexidade computacional entre todas as variações quando o número de classes em cada nível é menor que o número de atributos. Isso ocorre porque do segundo nível em diante, são utilizadas apenas as predições como entrada das redes [MLP](#). No primeiro nível, o custo computacional é o mesmo das outras variações. Do segundo nível em diante, a complexidade é dada por $O(W_l)$, com $W_l = (O_{l-1} + 1) \times H_l + (H_l + 1) \times O_l$.

Em todas as variações do método [HMC-LMLP](#), o custo total de treinamento de cada rede [MLP](#) associada a cada nível l é então definido como $O(W_l \times m_l \times n)$, sendo m_l o número de exemplos de treinamento classificados em classes pertencentes ao nível l , e n o número de ciclos de treinamento.

4.7 Considerações Finais

Este capítulo apresentou o método *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* ([HMC-LMLP](#)), uma proposta de utilização de redes neurais artificiais para classificação hierárquica multirrótulo, na qual uma rede [MLP](#) é associada a cada nível hierárquico e responsável pelas predições em seu respectivo nível. Foram propostas duas variações incrementais do método: utilizando as predições em um nível como as únicas entradas no próximo nível (HMC-LMLP-Labels), e utilizando as predições em um nível para complementar os vetores de atributos dos exemplos no próximo nível (HMC-LMLP-Predicted). Além disso, outras duas variações não incrementais foram apresentadas: utilizando as classes verdadeiras dos exemplos para complementar os vetores de atributos (HMC-LMLP-True), e utilizando apenas os exemplos sem as classes (HMC-LMLP-NoLabels). Foi também discutido como são

obtidas as predições finais e corrigidas as inconsistências de classificação, intrínsecas da estratégia adotada pelo método, que utiliza um classificador local por nível (LCL). O próximo capítulo apresenta os detalhes da proposta que utiliza algoritmos genéticos para a geração de regras de classificação hierárquicas multirrótulo.

Classificação Hierárquica Multirrótulo com um Algoritmo Genético

Este capítulo apresenta o método para classificação hierárquica multirrótulo utilizando Algoritmos Genéticos. Trata-se do método *Hierarchical Multi-Label Classification with a Genetic Algorithm* (HMC-GA), que é baseado na abordagem global para a geração de regras de classificação. O pseudo-código do módulo principal do método é apresentado no Algoritmo 3, que implementa um procedimento de cobertura sequencial de exemplos para evoluir os antecedentes das regras de classificação. Nesse procedimento de cobertura sequencial, os exemplos cobertos por uma regra são removidos do conjunto de treinamento, de maneira que novas regras possam ser geradas para cobrir os exemplos remanescentes. Ainda, no método proposto, o consequente de uma regra é obtido utilizando um procedimento determinístico que considera as classes atribuídas aos exemplos cobertos pela regra. Este capítulo está organizado da seguinte maneira: na Seção 5.1 é apresentado como um indivíduo do algoritmo é representado; a inicialização dos indivíduos é explicada na Seção 5.2; na Seção 5.3 é explicado todo o processo evolutivo que leva à construção dos antecedentes das regras; o procedimento utilizado para a construção dos consequentes das regras é apresentado na Seção 5.4; a complexidade computacional de algumas partes do método é apresentada na Seção 5.5; por fim, a Seção 5.6 apresenta as considerações finais do capítulo.

Algoritmo 3: Classificação Hierárquica Multirrótulo com um Algoritmo Genético.**procedimento** HMC-GA($D, G, p, minCob, maxCob, maxNaoCob, cr, mr, t, e, pt$)**Entrada:** conjunto de treinamento D número de gerações G tamanho da população p número mínimo de exemplos cobertos por uma regra $minCob$ número máximo de exemplos cobertos por uma regra $maxCob$ número máximo de exemplos não cobertos $maxNaoCob$ taxa de *crossover* cr taxa de mutação mr tamanho do torneio t número de indivíduos selecionados por elitismo e probabilidade de usar um termo pt **Saída:** conjunto de regras $regrasDescobertas$ 1 $regrasDescobertas \leftarrow \emptyset$ 2 **enquanto** $|D| > maxNaoCob$ **faça**3 $populacaoInicial \leftarrow geraPopulacao(D, p, pt)$ 4 $calculaFitness(populacaoInicial, D)$ 5 $populacaoAtual \leftarrow populacaoInicial$ 6 $melhorRegra \leftarrow$ melhor regra da $populacaoAtual$ segundo o *fitness*7 $j \leftarrow G$ 8 **repita**9 $novaPopulacao \leftarrow \emptyset$ 10 $novaPopulacao \leftarrow novaPopulacao \cup elite(populacaoAtual, e)$ 11 $pais \leftarrow selecaoTorneio(populacaoAtual, t, e, p)$ 12 $filhos \leftarrow crossoverUniformeDistancia(pais, cr)$ 13 $novaPopulacao \leftarrow novaPopulacao \cup filhos$ 14 $novaPopulacao \leftarrow mutacao(novaPopulacao, mr, pt)$ 15 $novaPopulacao \leftarrow operadorLocal(novaPopulacao, minCob, maxCob)$ 16 $populacaoAtual \leftarrow novaPopulacao$ 17 $calculaFitness(populacaoAtual, D)$ 18 $melhorRegra \leftarrow obtemMelhorRegra(populacaoAtual, melhorRegra)$ 19 $j \leftarrow j - 1$ 20 **até** $j > 0 \vee regraConverge()$;21 $regrasDescobertas \leftarrow regrasDescobertas \cup melhorRegra$ 22 remove de D os exemplos cobertos pela $melhorRegra$ 23 **retorna** $regrasDescobertas$

5.1 Representação dos Indivíduos

Os indivíduos do método **HMC-GA** são os antecedentes das regras de classificação. Um antecedente de uma regra é representado por um vetor de valores reais, no qual cada valor é decodificado em um componente do antecedente da regra, que pode ser um operador, um índice de atributo ou um valor de atributo. A Figura 5.1 apresenta a representação de um indivíduo no método **HMC-GA**.

Figura 5.1: Representação de um indivíduo no método HMC-GA.

Cada conjunto de quatro posições do vetor ilustrado na Figura 5.1 representa um teste sobre um atributo do conjunto de dados sendo utilizado para evolução das regras. Assim, o vetor antecedente tem tamanho igual a quatro vezes o número de atributos, e cada teste é codificado como um 4-tupla $\{\text{FLAG}, \text{OP}, \Delta_1, \Delta_2\}$. O gene **FLAG** pode ser tanto 0 quanto 1, indicando a presença (1) ou ausência (0) do respectivo teste no antecedente da regra, já que uma regra pode ser composta apenas por alguns atributos. O gene **OP** deve conter o índice de um dos possíveis operadores do teste, e os genes Δ_1 and Δ_2 são valores reais utilizados como condições dos testes.

Todos os testes que compõem o antecedente de uma regra são separados por uma cláusula E. Esses testes testam um dado atributo A_i de acordo com um valor Δ , utilizando um operador **OP**. No caso de atributos nominais, os operadores empregados podem ser $=$, $\neq$ e *in*. O operador *in* permite que um dado atributo categórico seja testado para verificar se seu valor pertence a um conjunto de valores. Para cada atributo categórico, todos os seus valores são indexados com valores numéricos, de modo a serem utilizados na representação dos indivíduos. Além disso, todos os possíveis conjuntos de dois ou mais valores formados pelos valores categóricos de um atributo também são indexados. Esse último procedimento é necessário para a utilização do operador *in*. Quando os operadores $=$ e $\neq$ são utilizados em um teste, Δ_2 recebe 0 e Δ_1 recebe o índice correspondente ao valor categórico utilizado no teste da condição. Quando o operador *in* é utilizado, Δ_2 recebe 0 e Δ_1 recebe o índice correspondente ao conjunto de valores categóricos utilizado no teste da condição.

No caso de atributos numéricos, podem ser utilizados os operadores $\geq$ e $\leq$. Além disso, é possível que um atributo numérico seja testado considerando um operador composto, para verificar se seu valor pertence a um dado intervalo, ou seja, $\Delta_1 \leq A_i \leq \Delta_2$. Nos casos em que são utilizados operadores simples (não compostos), os valores de Δ_1 e Δ_2 são escolhidos de acordo com o operador. Se o operador for $\leq$, Δ_1 recebe o valor 0 e Δ_2 recebe o valor de teste da condição. Para o operador $\geq$, Δ_2 recebe 0 e Δ_1 recebe o valor de teste da condição.

Todos os operadores possíveis, numéricos e categóricos, e também os possíveis valores categóricos de um atributo, são previamente indexados e possuem índices fixos. Assim um teste pode ser facilmente executado por meio da recuperação dos índices e utilização do operador e valores correspondentes. A Figura 5.2 ilustra um exemplo desse processo.

Figura 5.2: Exemplo de indexação de valores categóricos e operadores do método HMC-GA.

Após construído o antecedente de uma regra, esse deve ser capaz de classificar um dado exemplo em um conjunto de classes, respeitando as restrições da estrutura hierárquica na qual as classes estão organizadas. Assim, uma regra de classificação para problemas hierárquicos multirrótulo do método **HMC-GA** deve ser estruturada como (lembrando que para atributos numéricos, testes compostos do tipo $\Delta_1 \leq A_i \leq \Delta_2$ também são possíveis):

$$\begin{aligned}
 & \text{SE } (A_1 \text{ OP } \Delta) \text{ E } \dots \text{ E } (A_m \text{ OP } \Delta) \rightarrow \text{Antecedente} \\
 & \text{ENTÃO} \\
 & \{C_1, C_{1.3}, C_{1.4}, C_{1.4.2}\} \rightarrow \text{Consequente}
 \end{aligned}$$

5.2 Inicialização da População

A população é iniciada por meio de um procedimento de semeadura (Algoritmo 3, linha 3), no qual um exemplo de treinamento é aleatoriamente selecionado e seus atributos são utilizados para criar um indivíduo. Após um exemplo ser selecionado, cada um de seus atributos possui uma probabilidade pt de ser utilizado no antecedente correspondente ao indivíduo sendo gerado. Isso significa que existe uma probabilidade pt do FLAG pertencente a 4-tupla correspondente a um atributo A_i receber o valor 1.

Após o FLAG de uma 4-tupla correspondente a um atributo receber um valor (1 para ativado e 0 para desativado), são preenchidas as posições do indivíduo correspondentes aos operadores e valores do teste sobre o atributo. O operador utilizado é selecionado aleatoriamente dependendo se o atributo em questão é numérico ou categórico, e a posição correspondente ao operador (OP) recebe um índice, como mostrado na Figura 5.2.

Após a escolha do operador, se o mesmo corresponder a uma operação sobre um atributo categórico, a escolha do(s) valor(es) utilizado(s) no teste depende do operador escolhido. Se o operador for $=$ ou $\neq$, a posição Δ_1 recebe o índice correspondente ao valor do atributo no exemplo sendo utilizado, e a posição Δ_2 recebe o valor 0. Se o operador for in , a posição Δ_1

recebe o índice correspondente a um dos conjuntos de valores que contenham o valor do atributo no exemplo, e a posição Δ_2 recebe o valor 0. Exemplificando esse último procedimento, se o atributo no exemplo possuir o valor A, e os possíveis valores do atributo no conjunto de dados forem A, B e C, a posição Δ_1 recebe o índice correspondente a um dos possíveis conjuntos de valores que contém A: {A, B}, {A, C} e {A, B, C}. O conjunto de valores utilizado é escolhido aleatoriamente. A indexação dos valores categóricos é feita como ilustrado na Figura 5.2.

Se o operador escolhido corresponder a uma operação sobre um atributo numérico, o preenchimento dos valores da 4-tupla correspondente ao atributo é mais simples. No caso do operador $\geq$, a posição Δ_1 recebe o valor do atributo no exemplo, e a posição Δ_2 recebe 0. Se o operador for o $\leq$, a posição Δ_2 recebe o valor do atributo no exemplo, e a posição Δ_1 recebe 0. Se o operador escolhido realizar um teste para verificar se o valor do atributo encontra-se em um intervalo ($\Delta_1 \leq A_i \leq \Delta_2$), os valores de Δ_1 e Δ_2 são escolhidos aleatoriamente de maneira que o valor do atributo no exemplo satisfaça a condição do teste.

O procedimento de sementeira é repetido até que seja formada uma população com o número desejado de indivíduos. Esse procedimento garante que cada indivíduo (antecedente da regra) cubra pelo menos um exemplo de treinamento. Um antecedente cobre um exemplo se todos os seus termos ativos são satisfeitos pelos correspondentes atributos do exemplo. A Figura 5.3 exemplifica o procedimento de sementeira aplicado a um exemplo.

Figura 5.3: Exemplo do procedimento de sementeira do método HMC-GA.

5.3 Evolução

O processo de evolução é iniciado salvando as melhores e regras da população atual. Um conjunto de $p - e$ regras pais (sendo p o tamanho da população e e o número de indivíduos selecionados no elitismo) é então selecionado utilizando seleção por torneio (Algoritmo 3, linha 11). Em seguida, essas regras pais são submetidas a um procedimento de *crossover* para a geração de um conjunto de $p - e$ regras filhas (Algoritmo 3, linha 12). De maneira a especializar as regras na classificação de um conjunto de classes, é proposto um operador de *crossover* que considera as distâncias entre os vetores médios de classes das regras. Esse *crossover* especializado é apresentado no Algoritmo 4.

Algoritmo 4: *Crossover* uniforme baseado em distância.

procedimento *crossover*UniformeDistancia($pais, cr$)

Entrada: conjunto de pais $pais$
taxa de *crossover* cr

Saída: conjunto de filhos $filhos$

```

1  $filhos \leftarrow \emptyset$ 
2 enquanto  $|pais| > 0$  faça
3    $pai_1 \leftarrow$  primeira regra de  $pais$ 
4    $pais \leftarrow pais - pai_1$ 
5    $pai_2 \leftarrow$  paiMaisProximo( $pai_1, pais$ )
6    $pais \leftarrow pais - pai_2$ 
7    $filhos \leftarrow filhos \cup$  crossoverUniforme( $pai_1, pai_2, cr$ )
8 retorna  $filhos$ 
```

O vetor médio de classes de uma regra r é representado por um vetor $\bar{v}_r$ de tamanho $|C|$, sendo C o conjunto de todas as classes da hierarquia. O cálculo do i -ésimo componente de $\bar{v}_r$ é apresentado na Equação 5.1, em que $S_{r,i}$ é o conjunto de todos os exemplos de treinamento cobertos pela regra r que são classificados na classe i , e S_r é o conjunto de todos os exemplos de treinamento cobertos pela regra r . Dessa forma, cada posição $\bar{v}_{r,i}$ contém a proporção de exemplos cobertos por r que são classificados na classe i (valores entre 0 e 1).

$$\bar{v}_{r,i} = \frac{|S_{r,i}|}{|S_r|} \quad (5.1)$$

O operador de *crossover* especializado recebe como entrada uma lista de $p - e$ regras pais. Uma regra é então removida da lista, e as distâncias Euclidianas entre o vetor médio de classes dessa regra e os vetores médios de classes de todas as outras regras da lista são calculadas. Quanto menor a distância Euclidiana entre os vetores médios de classes de duas regras, mais próximas são as regras. O cálculo da distância Euclidiana é apresentado na Equação 5.2. A regra mais próxima é então removida da lista, e então as duas regras pais selecionadas são submetidas a uma operação convencional de *crossover* uniforme para a geração das regras filhas. Essa operação tem o objetivo de aplicar o operador de *crossover* em regras que cobrem exemplos próximos no espaço de busca, ou seja, exemplos que são classificados em um conjunto

de classes igual ou similar. Esse procedimento de seleção de duas regras para cruzamento é repetido até uma nova geração completa de regras ser obtida.

$$distanciaEuclidiana(\bar{v}_1, \bar{v}_2) = \sqrt{\sum_{i=1}^{|C|} w_i \times (\bar{v}_{1,i} - \bar{v}_{2,i})^2} \quad (5.2)$$

Na Equação 5.2, w_i corresponde ao peso associado a i -ésima classe da hierarquia, e $\bar{v}_{1,i}$ e $\bar{v}_{2,i}$ são os valores associados a i -ésima posição dos vetores médios de classes $\bar{v}_1$ e $\bar{v}_2$. Pesos foram associados a todas as classes porque, no contexto de classificação hierárquica, similaridades entre classes localizadas em níveis mais próximos à raiz são mais importantes do que similaridades entre classes localizadas em níveis mais profundos (Vens et al., 2008).

O esquema de pesos utilizado no método HMC-GA é o mesmo utilizado em (Vens et al., 2008). Após testar diferentes esquemas, os autores descobriram que o melhor deles era definido de acordo com a Equação 5.3. Nesse esquema, o peso w_0 associado a uma classe do primeiro nível foi definido como 0.75, e o peso de uma classe i é recursivamente definido como a multiplicação de w_0 e o peso médio de todas as suas classes pais P_i .

$$w_i = w_0 \times \sum_{j=1}^{P_i} w(p_j) / P_i \quad (5.3)$$

Após a geração de um novo conjunto de regras filhas, uma operação de mutação é aplicada (Algoritmo 3, linha 14) a uma porcentagem mr de indivíduos. Cada um desses indivíduos pode sofrer uma mutação na posição correspondente ao FLAG (ter um teste removido ou incluído com uma probabilidade pt , o que corresponde respectivamente a 0 ou 1 na posição correspondente ao FLAG) ou uma operação de generalização/restrição. As operações de generalização/restrição modificam os valores de Δ dos testes das regras de maneira a tornar as regras mais gerais ou mais específicas. Os valores Δ podem ser aumentados (Equação 5.4) ou diminuídos (Equação 5.5) de acordo com um fator de correção (*fator*) gerado aleatoriamente num intervalo entre 0 e 1. O Algoritmo 5 apresenta o operador de mutação do método.

$$\Delta_i = \Delta_i + (fator \times \Delta_i) \quad (5.4)$$

$$\Delta_i = \Delta_i - (fator \times \Delta_i) \quad (5.5)$$

Como pode ser observado no Algoritmo 5, linha 1, os indivíduos que sofrerão mutação são inicialmente selecionados de acordo com uma taxa de mutação mr . Cada um dos indivíduos selecionados possui 50% de chance de sofrer mutação no gene correspondente ao FLAG, e 50% de chance de sofrer uma generalização ou restrição de acordo com o operador do teste que sofrerá mutação. Se a mutação do FLAG for escolhida, todas as 4-tuplas do indivíduo são percorridas, e os genes correspondentes ao FLAG de cada teste são trocados de 0 para 1, ou de 1 para 0, de acordo com uma probabilidade pt (Algoritmo 5, linha 6).

É importante notar que a probabilidade pt é a mesma probabilidade utilizada no procedi-

mento de semente da inicialização da população. Geralmente, um valor baixo de pt é utilizado no procedimento de semente, pois um valor alto resultaria na criação de regras com muitos testes ativos, que consequentemente cobririam poucos ou apenas o exemplo utilizado como semente. Essa mesma lógica é aplicada para a mutação dos FLAGS, pois a utilização de um valor alto de pt ativaria muitos testes no antecedente, resultando em uma regra que cobriria poucos ou nenhum exemplo.

No caso de a mutação do FLAG não ser a operação escolhida, todos os testes ativos do indivíduo são percorridos para que seja realizada uma operação de generalização ou restrição. Se o atributo de um teste for numérico, é feita uma escolha aleatória entre as operações de generalização e restrição (Algoritmo 5, linha 10). Se a operação de generalização for escolhida, ela é aplicada de acordo com o operador presente no teste sendo generalizado, escolhendo os valores adequados para serem subtraídos ou somados aos valores dos genes correspondentes a Δ_1 e Δ_2 . O mesmo ocorre se a operação de restrição for escolhida. Os Algoritmos 6 e 7 apresentam, respectivamente, as operações de generalização e restrição em atributos numéricos.

Algoritmo 5: Operador de mutação do método HMC-GA.

procedimento *mutacao(individuos, mr, pt)*

Entrada: conjunto de indivíduos *individuos*

taxa de mutação *mr*

probabilidade de usar um termo *pt*

Saída: conjunto de indivíduos com mutação *individuosMutacao*

```

1 individuosMutacao  $\leftarrow$  selecionaMutacao(individuos, mr)
2 restanteIndividuos  $\leftarrow$  individuos – individuosMutacao
3 para cada individuo de individuosMutacao faça
4   numero  $\leftarrow$  número aleatório no intervalo [0, 1]
5   se numero  $\leq$  0.5 então
6     individuo  $\leftarrow$  mutacaoFlag(individuo, pt)
7   senão
8     para cada testeAtivo de individuo faça
9       se atributo for numérico então
10        operacao  $\leftarrow$  0 ou 1 escolhidos aleatoriamente
11        caso operacao = 0
12          testeAtivo  $\leftarrow$  generalizacaoNumerico(testeAtivo)
13        caso operacao = 1
14          testeAtivo  $\leftarrow$  restricaoNumerico(testeAtivo)
15        senão
16          testeAtivo  $\leftarrow$  generalizacaoRestricaoCategorico(testeAtivo)
17        individuo  $\leftarrow$  atualizaTeste(testeAtivo)
18 individuosMutacao  $\leftarrow$  individuosMutacao  $\cup$  restanteIndividuos
19 retorna individuosMutacao

```

Se as operações de generalização ou restrição forem ser aplicadas em um atributo categórico, a operação aplicada depende do operador do teste (Algoritmo 5, linha 16). Se o operador do teste for o =, o teste pode ser apenas generalizado. Essa generalização ocorre pela substituição

Algoritmo 6: Operador de generalização de testes em atributos numéricos.

procedimento *generalizacaoNumerico*(*testeAtivo*)

Entrada: teste pertencente a um indivíduo *testeAtivo*
Saída: teste generalizado *testeAtivoGeneralizado*

```

1  random  $\leftarrow$  número aleatório no intervalo [0, 1]
2  caso operador = " $\leq$ "
3      fator  $\leftarrow$  testeAtivo $\Delta_2$   $\times$  random
4      se testeAtivo $\Delta_2$  = 0 então
5           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + random
6      senão se testeAtivo $\Delta_2$  > 0 então
7           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + fator
8      senão
9           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - fator
10 caso operador = " $\geq$ "
11     fator  $\leftarrow$  testeAtivo $\Delta_1$   $\times$  random
12     se testeAtivo $\Delta_1$  = 0 então
13          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - random
14     senão se testeAtivo $\Delta_1$  > 0 então
15          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - fator
16     senão
17          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + fator
18 caso operador = " $\leq \leq$ "
19     fatorInferior  $\leftarrow$  testeAtivo $\Delta_1$   $\times$  random
20     fatorSuperior  $\leftarrow$  testeAtivo $\Delta_2$   $\times$  random
21     se testeAtivo $\Delta_1$  = 0 então
22          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - random
23     senão se testeAtivo $\Delta_1$  > 0 então
24          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - fatorInferior
25     senão
26          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + fatorInferior
27     se testeAtivo $\Delta_2$  = 0 então
28          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + random
29     senão se testeAtivo $\Delta_2$  > 0 então
30          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + fatorSuperior
31     senão
32          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - fatorSuperior
33 testeAtivoGeneralizado  $\leftarrow$  testeAtivo
34 retorna testeAtivoGeneralizado

```

Algoritmo 7: Operador de restrição de testes em atributos numéricos.**procedimento** *restricaoNumerico*(*testeAtivo*)**Entrada:** teste pertencente a um indivíduo *testeAtivo***Saída:** teste restringido *testeAtivoRestringido*

```

1  random  $\leftarrow$  número aleatório no intervalo  $[0, 1]$ 
2  caso operador = " $\leq$ "
3      fator  $\leftarrow$  testeAtivo $\Delta_2$   $\times$  random
4      se testeAtivo $\Delta_2$  = 0 então
5           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - random
6      senão se testeAtivo $\Delta_2$  > 0 então
7           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - fator
8      senão
9           $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + fator
10 caso operador = " $\geq$ "
11     fator  $\leftarrow$  testeAtivo $\Delta_1$   $\times$  random
12     se testeAtivo $\Delta_1$  = 0 então
13          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + random
14     senão se testeAtivo $\Delta_1$  > 0 então
15          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + fator
16     senão
17          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - fator
18 caso operador = " $\leq \leq$ "
19     fatorInferior  $\leftarrow$  testeAtivo $\Delta_1$   $\times$  random
20     fatorSuperior  $\leftarrow$  testeAtivo $\Delta_2$   $\times$  random
21     se testeAtivo $\Delta_1$  = 0 então
22          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + random
23     senão se testeAtivo $\Delta_1$  > 0 então
24          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  + fatorInferior
25     senão
26          $\lfloor$  testeAtivo $\Delta_1$   $\leftarrow$  testeAtivo $\Delta_1$  - fatorInferior
27     se testeAtivo $\Delta_2$  = 0 então
28          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - random
29     senão se testeAtivo $\Delta_2$  > 0 então
30          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  - fatorSuperior
31     senão
32          $\lfloor$  testeAtivo $\Delta_2$   $\leftarrow$  testeAtivo $\Delta_2$  + fatorSuperior
33 testeAtivoRestringido  $\leftarrow$  testeAtivo
34 retorna testeAtivoRestringido

```


do operador = pelos operadores $\neq$ ou *in*. Nesse caso, é feita uma escolha aleatória entre esses dois operadores. Se o operador $\neq$ for escolhido, ele simplesmente substitui o operador = no teste. Se o operador *in* for escolhido, é necessário um procedimento adicional à troca de = por *in*. Além dessa troca, o valor categórico atual da condição do teste deve ser substituído por um conjunto de valores que contenha o valor atual. Se houver mais de um conjunto possível, é feita uma escolha aleatória entre todos eles.

No caso do operador presente no teste ser o $\neq$, o teste pode ser restringido se o operador $\neq$ for substituído pelo operador =. Ainda, pode-se substituir o operador $\neq$ pelo operador *in*. Nesse caso, o teste pode ser generalizado ou restringido dependendo dos valores escolhidos para compor o teste. Exemplificando, suponha que a condição do teste seja *Atributo_i $\neq$ B*, e que os possíveis valores do *Atributo_i* sejam A, B, C e D. Se a condição do teste for substituída por *Atributo_i in {A, C}*, o teste é restringido. Entretanto, se a condição do teste for substituída por *Atributo_i in {A, B, C, D}*, o teste é generalizado.

Se o operador presente no teste for o *in*, o teste também pode ser generalizado ou restringido dependendo dos novos valores escolhidos para o teste. O conjunto de valores atual do teste pode ser substituído por um conjunto com mais ou menos valores, e também por um único valor. Se o conjunto atual de valores for substituído por um único valor, o operador *in* é substituído pelo operador =. O Algoritmo 8 apresenta o procedimento de generalização/restrrição de testes aplicados a atributos categóricos.

Após a aplicação dos operadores de mutação e geração de um novo conjunto de indivíduos, um operador de busca local é aplicado a esse conjunto (Algoritmo 3, linha 15). Esse operador visa garantir que os antecedentes das regras geradas cubram entre um número mínimo e máximo de exemplos, fazendo com que as regras não sejam muito específicas e nem muito gerais. O número mínimo (*minCobertos*) e máximo (*maxCobertos*) de exemplos cobertos por uma regra são parâmetros especificados pelo usuário. Esse procedimento de busca local é apresentado no Algoritmo 9.

Alguns detalhes devem ser observados no Algoritmo 9. O primeiro deles diz respeito ao número máximo de tentativas executadas para a convergência de uma regra. Se depois de um número *MAX* de tentativas, os dois critérios *minCobertos* e *maxCoberto* não forem satisfeitos, o antecedente final da regra é o obtido na última tentativa. Essa condição de parada é necessária, pois pode ser que uma regra nunca satisfaça os critérios desejados.

O segundo detalhe a ser observado diz respeito ao procedimento executado quando uma regra é muito específica (Algoritmo 9, linha 6). Se o número de testes ativos no antecedente da regra é maior que 1, um dos testes pode ser aleatoriamente removido de modo a deixar a regra mais geral. No entanto, pode ocorrer de uma regra possuir apenas 1 teste ativo em seu antecedente, e ainda assim ser muito específica. Nesse caso, o teste em questão é substituído por outros teste escolhido aleatoriamente, e esse teste é generalizado dependendo se o atributo correspondente ao teste é numérico ou categórico.

Algoritmo 8: Operador de restrição/generalização de testes em atributos categóricos.

procedimento *restricaoGeneralizacaoCategorico(testeAtivo)*

Entrada: teste pertencente a um indivíduo *testeAtivo*

Saída: teste restringido/generalizado *testeAtivoRestrGener*

```

1 caso operador = "="
2   random ← número aleatório no intervalo [0, 1]
3   se random < 0.5 então
4     | testeAtivoOP ← 1 // 1 corresponde ao operador ≠
5   senão
6     | testeAtivoOP ← 2 // 2 corresponde ao operador in
7     | conjuntoValores ← conjunto de valores, escolhido aleatoriamente, que contém o
      | valor atual do teste
8     | testeAtivoΔ1 ← índice do conjunto de valores escolhido
9 caso operador = "≠"
10  | random ← número aleatório no intervalo [0, 1]
11  | se random < 0.5 então
12    | testeAtivoOP ← 0 // 0 corresponde ao operador =
13  | senão
14    | testeAtivoOP ← 2 // 2 corresponde ao operador in
15    | conjuntoValores ← conjunto de valores escolhido aleatoriamente
16    | testeAtivoΔ1 ← índice do conjunto de valores escolhido
17 caso operador = "in"
18  | conjuntoValores ← conjunto de valores, ou valor único, escolhido aleatoriamente,
      | diferente do atual
19  | testeAtivoΔ1 ← índice do conjunto de valores, ou valor único, escolhido
20  | se valor único foi escolhido então
21    | testeAtivoOP ← 0 // 0 corresponde ao operador =
22 testeAtivoRestrGener ← testeAtivo
23 retorna testeAtivoRestrGener

```

Algoritmo 9: Operador de busca local do método HMC-GA.

procedimento operadorLocal(*populacao*, *minCobertos*, *maxCobertos*)

Entrada: população de indivíduos *populacao*

número mínimo de exemplos cobertos por uma regra *minCobertos*

número máximo de exemplos cobertos por uma regra *maxCobertos*

Saída: população modificada *novaPopulacao*

```

1 para cada individuo de populacao faça
2   maxTentativas  $\leftarrow$  0
3   convergencia  $\leftarrow$  0
4   enquanto convergencia = 0 E maxTentativas < MAX faça
5     maxTentativas  $\leftarrow$  maxTentativas + 1
6     se número de exemplos cobertos < minCobertos então
7       caso número de testes ativos > 1
8          $\lfloor$  individuo  $\leftarrow$  removeTesteAtivo(individuo)
9       caso número de testes ativos = 1
10        substitui o teste do indivíduo por um novoTeste escolhido aleatoriamente
11        se atributo numérico então
12           $\lfloor$  novoTeste  $\leftarrow$  generalizacaoNumerico(teste)
13        senão
14           $\lfloor$  novoTeste  $\leftarrow$  generalizacaoCategorico(teste)
15         $\lfloor$  individuo  $\leftarrow$  atualizaTeste(novoTeste)
16      senão se número de exemplos cobertos > maxCobertos então
17        se número de testes ativos < número de atributos do problema então
18           $\lfloor$  individuo  $\leftarrow$  adicionaTesteAtivo(individuo)
19      senão
20         $\lfloor$  convergencia  $\leftarrow$  1
21 novaPopulacao  $\leftarrow$  populacao
22 retorna novaPopulacao

```

Um terceiro detalhe é observado quando uma regra é muito geral Algoritmo 9, linha 16). Nesse caso, um teste deve ser adicionado ao antecedente da regra, de maneira a torna-la mais específica. No entanto, pode ocorrer de o antecedente de uma regra possuir todos os seus testes ativos (utiliza todos os atributos da base de dados) e ainda assim ser muito geral. Nesse caso, nenhum procedimento é executado, já que uma regra assim é rara e pode contribuir positivamente para a geração das regras futuras.

Ao final da geração de uma nova população de indivíduos, a melhor regra da população é salva (Algoritmo 3, linha 18). Para a escolha da melhor regra, dois fatores são considerados, o *fitness* da regra, e se a mesma cobre o número mínimo de exemplos especificado em *minCobertos*. Inicialmente, quando a primeira geração de indivíduos é criada, a melhor regra é salva de acordo com seu *fitness*. A partir da segunda geração, todas as regras geradas são ordenadas de acordo com seus valores de *fitness* e comparadas uma a uma com a melhor regra da geração anterior. Se uma nova regra tiver o valor de *fitness* maior do que o valor de *fitness* da melhor regra da geração anterior, e também cobrir um número de exemplos maior que *minCobertos*, ela será escolhida a melhor regra até o momento. O processo evolutivo (Algoritmo 3, linhas 8 a 19) é executado até que o número máximo de gerações seja atingido, ou até a convergência da população de regras. A população converge quando uma regra permanece sendo a melhor após um número específico de gerações.

Ao final do processo evolutivo, a melhor regra obtida é salva em um conjunto de regras (Algoritmo 3, linha 21), e os exemplos cobertos por essa regra são removidos do conjunto de treinamento (Algoritmo 3, linha 22). O processo de evolução é então reiniciado com a geração de um novo conjunto de regras iniciais. Esse processo de cobertura sequencial de exemplos é repetido até que todos, ou quase todos (parâmetro *maxNaoCob* do Algoritmo 3), os exemplos de treinamento sejam cobertos. Se sobraem exemplos não cobertos no conjunto de treinamento, os mesmos são classificados em uma regra padrão. Essa regra simplesmente classifica esses exemplos no vetor médio de todas as classes do conjunto de treinamento.

O vetor médio $\bar{\mathbf{v}}$ de todas as classes do conjunto de treinamento é obtido utilizando todos os vetores de classes de todos os exemplos de treinamento. Cada exemplo de treinamento i do conjunto de treinamento S possui suas classes representadas em um vetor $\mathbf{v}_i$ de $|C|$ posições, com C sendo o conjunto de classes do problema. Se um exemplo i pertence a uma classe j , a posição v_{ij} do vetor $\mathbf{v}_i$ recebe o valor 1, e 0 caso contrário. Cada posição $\bar{v}_j$ do vetor médio de classes $\bar{\mathbf{v}}$ do conjunto de treinamento S é então obtido por $(\sum_i^{|S|} v_{ij})/|S|$.

5.4 Construção dos Consequentes e Cálculo do Fitness

Na Equação 5.1, foi mostrado como é construído o vetor médio de classes $\bar{\mathbf{v}}_r$ de uma regra, obtido utilizando os vetores de classes dos exemplos cobertos pela regra. Esse vetor médio também pode ser interpretado como o consequente da regra. Assim, cada posição do vetor consequente de uma regra é um valor contínuo no intervalo $[0,1]$. Como resultado, o valor do i -ésimo componente do vetor consequente de uma regra representa a probabilidade de um

exemplo que satisfaz seu antecedente pertencer a classe i da hierarquia.

Uma vez construídos os consequentes das regras, essas são avaliadas por meio de uma função de *fitness*. A função de *fitness* utilizada aqui é o ganho de variância (Vens et al., 2008; Otero et al., 2010). O ganho de variância é maior para regras que cobrem um conjunto mais homogêneo de exemplos e deixam sem cobertura também um conjunto mais homogêneo de exemplos, ou seja, regras que particionam o conjunto de treinamento em conjuntos mais homogêneos. Além disso, o ganho de variância lida diretamente com dados hierárquicos multirrótulo, levando em consideração os relacionamentos e similaridades entre as classes (Otero et al., 2010). O ganho de variância é apresentado na Equação 5.6.

$$\text{ganhoVariância}(r, S) = \text{variância}(S) - \frac{|S_r|}{|S|} \times \text{variância}(S_r) - \frac{|S_{\neg r}|}{|S|} \times \text{variância}(S_{\neg r}) \quad (5.6)$$

Como observado na Equação 5.6, o conjunto de treinamento S é dividido em dois subconjuntos: o conjunto de exemplos cobertos pela regra r , denominado S_r , e o conjunto de exemplos não cobertos pela regra r , denominado $S_{\neg r}$. O ganho de variância da regra r é então calculado com relação ao conjunto S . Além disso, esse cálculo envolve a variância dos conjuntos de exemplos cobertos e não cobertos pela regra. Essa variância é definida como a distância quadrática média entre o vetor de classes de cada exemplo e o vetor médio de classes de todo conjunto de exemplos (S , S_r ou $S_{\neg r}$) sendo utilizado. O cálculo da variância de um conjunto de exemplos S é apresentado na Equação 5.7. A distância utilizada é a distância Euclidiana ponderada, apresentada anteriormente na Equação 5.2.

$$\text{variância}(S) = \frac{\sum_{k=1}^{|S|} \text{distanciaEuclidiana}(\mathbf{v}_k, \bar{\mathbf{v}})^2}{|S|} \quad (5.7)$$

5.5 Complexidade Computacional

A análise de complexidade computacional de algoritmos evolutivos geralmente não é feita, pois apesar de estudos teóricos (He e Yao, 2003) sugerirem o uso de cadeias de Markov para esse fim, não se trata de uma tarefa de análise simples. Assim, esta seção apresenta a complexidade computacional dos procedimentos mais críticos do método HMC-GA.

São considerados com maior custo computacional o procedimento de semeadura utilizado para a criação da população inicial (Seção 5.2), o operador de busca local que visa evitar que as regras sejam muito específicas ou muito gerais (Algoritmo 9), e o cálculo do *fitness* das regras (Seção 5.4). Além desses procedimentos, existem outros procedimentos executados durante a evolução do algoritmo genético, como a operação de *crossover*, que também consomem tempo. No entanto, eles também utilizam funções que são utilizadas no cálculo do *fitness* e na busca local, como o cálculo da distância entre dois vetores médios de classes, ou utilizam funções que já foram calculadas previamente, como a verificação de quais exemplos são cobertos por uma regra. Todas essas funções serão analisadas dentro das operações de busca local e cálculo

do *fitness*.

Inicialização da população

A inicialização da população é considerada crítica porque todos os atributos do exemplo selecionado como semente para dar origem a uma regra devem ser percorridos. Ainda, para cada atributo do exemplo, existem quatro posições no vetor representando o indivíduo sendo construído. Assim, sendo $|A|$ o número de atributos de um conjunto de dados, e pop o número de indivíduos da população inicial, a complexidade computacional do procedimento de semeadura é dado por $O(|A| \times 4 \times pop)$.

Operador de busca local

O procedimento de busca local é executado enquanto a convergência de uma regra não é alcançada. Essa convergência é alcançada quando o número de exemplos cobertos por uma regra encontra-se entre um valor mínimo e um valor máximo de exemplos. Existem dois cenários para o pior caso nessa busca. No primeiro cenário, uma regra tem todos os atributos ativados, sendo muito específica (cobrindo um número muito pequeno de exemplos). Nesse caso ela terá um número de termos ativos igual a $|A|$. O pior caso ocorre quando $|A| - 1$ atributos do antecedente da regra necessitam ser desativados ($FLAG = 0$) para que a regra cubra um número desejado de exemplos, entre um mínimo e um máximo. Assim, todos os atributos necessitam ser percorridos, resultando em complexidade computacional igual a $O(|A|)$. O segundo cenário ocorre quando a regra tem apenas um atributo ativado, sendo muito geral, e todos os seus atributos necessitam ser ativados ($FLAG = 1$) para que a regra cubra o número desejado de exemplos. Nesse caso, também todos os atributos devem ser percorridos, sendo a complexidade igual a $O(|A|)$.

Além de percorrer os atributos de uma regra, toda vez que um atributo é ativado ou desativado, é necessário que seja computado o número de exemplos cobertos pela regra. Para isso, é necessário que cada atributo ativo de uma regra seja comparado com o respectivo atributo de todos os exemplos. Considere o cenário de pior caso no qual uma regra tem apenas um atributo ativado, e necessita que todos os seus atributos sejam ativados para que ela cubra o número desejado de exemplos. Assim, após a ativação do segundo atributo, os dois atributos da regra devem ser comparados com os respectivos atributos dos exemplos. Com $|X|$ sendo o número total de exemplos, são realizadas $|X| \times 2$ comparações. Ainda, como a comparação envolve a verificação da 4-tupla correspondente ao atributo na regra, esse número de comparações deve ser multiplicado por quatro, resultando em $|X| \times 2 \times 4$ comparações. Assim, o custo total das comparações a medida que atributos do exemplo vão sendo ativados é dado por:

$$\begin{aligned}
Custo_{comp} &= 2 \times 4 \times |X| + 3 \times 4 \times |X| + \dots + |A| \times 4 \times |X| \\
&= |X| \times 4 \times \sum_{i=2}^{|A|} i \\
&= |X| \times 4 \times \left(\frac{|A|(|A| + 1)}{2} - 1 \right)
\end{aligned}$$

Sendo essa equação quadrática do número de atributos $|A|$, a complexidade computacional da busca local é dada por $O(|X| \times 4 \times |A|^2)$.

Cálculo do fitness

O cálculo do *fitness* de uma regra consiste no cálculo das variâncias de todos os exemplos de treinamento, do conjunto de exemplos cobertos, e do conjunto de exemplos não cobertos pela regra. A variância de um conjunto de exemplos consiste no cálculo do vetor médio de classes dos exemplos, e depois, do cálculo das Distâncias Euclidianas entre os vetores de classes de todos os exemplos e o vetor médio de classes.

O vetor médio de classes de um conjunto de exemplos é dado pela soma de todos os valores v_j dos vetores de classes $\mathbf{v}$ de cada exemplo, dividido pelo número total de exemplos do conjunto. Assim, sendo $|C|$ o número de classes do problema, o número de operações necessárias para o cálculo do vetor médio de classes é dado por $|X| \times |C|$. Sendo S_r e $S_{\neg r}$ os conjuntos de exemplos cobertos e não cobertos por uma regra r , os cálculos dos vetores médios desses respectivos conjuntos são dados por $|S_r| \times |C|$ e $|S_{\neg r}| \times |C|$.

Após a obtenção dos vetores médios de classes, são calculadas as Distâncias Euclidianas entre cada vetor de classes dos exemplos de cada conjunto de exemplos ($X, S_r, S_{\neg r}$) e os respectivos vetores médios de classes desses conjuntos. A Distância Euclidiana entre dois vetores de classes requer $|C|$ operações. Portanto, o cálculo da Distância Euclidiana envolvendo todos os exemplos de cada conjunto de exemplos requer, respectivamente, $|X| \times |C|$, $|S_r| \times |C|$ e $|S_{\neg r}| \times |C|$ operações.

Assim, as variâncias dos conjuntos X , S_r e $S_{\neg r}$ possuem, respectivamente, complexidade igual a $O(2 \times |X| \times |C|)$, $O(2 \times |S_r| \times |C|)$ e $O(2 \times |S_{\neg r}| \times |C|)$.

5.6 Considerações Finais

Este capítulo apresentou o método *Hierarchical Multi-Label Classification with a Genetic Algorithm* (**HMC-GA**), um método baseado na abordagem global que induz regras de classificação hierárquicas multirrotulo. O processo evolutivo é utilizado para a geração dos antecedentes das regras, e seus respectivos consequentes são gerados por meio de um processo determinístico que utiliza as classes dos exemplos cobertos pelos antecedentes. Ainda, o método utiliza um procedimento de cobertura sequencial de exemplos, no qual ao final de várias gerações de um

processo completo de evolução de regras, a melhor regra é salva, e os exemplos cobertos por ela são removidos do conjunto de treinamento. Todos os detalhes do método foram apresentados, desde a representação dos indivíduos e o procedimento para geração da população inicial, até os detalhes envolvidos nos operadores utilizados no processo evolutivo dos antecedentes e construção dos respectivos consequentes. O próximo capítulo apresenta os experimentos realizados envolvendo os dois métodos propostos nesta pesquisa de Doutorado.

Experimentos

Neste capítulo são apresentados os experimentos realizados. As medidas de avaliação utilizadas e os métodos da literatura utilizados nos experimentos para fins de comparação são apresentados, respectivamente, nas Seções 6.1 e 6.2; a seção 6.3 descreve os conjuntos de dados utilizados; na Seção 6.4 são apresentados e discutidos os experimentos realizados com o método HMC-LMLP, e a Seção 6.5 contém os experimentos com o método HMC-GA; na Seção 6.6, são apresentados gráficos comparando os desempenhos de todos os métodos utilizados nos experimentos; por fim, a Seção 6.7 é dedicada às considerações finais do Capítulo.

6.1 Medidas de Avaliação

Como visto nos Capítulos 4 e 5, os métodos HMC-LMLP e HMC-GA possuem a característica comum de que suas saídas, para cada exemplo, são formadas por um vetor no qual cada posição corresponde à predição para uma classe. Ainda, cada posição possui um valor real entre 0 e 1. Esse é o mesmo tipo de saída fornecida pelos métodos da literatura utilizados nos experimentos. Assim, de maneira a se obter as predições finais de todos os métodos, um valor de limiar deve ser utilizado. Quando classificando um exemplo, se o valor de saída correspondente a uma dada classe for igual ou superior a um dado limiar, a classe é atribuída ao exemplo. Caso contrário, a classe não é atribuída ao exemplo. Atribuir uma classe a um exemplo significa atribuir o valor 1 à posição correspondente a essa classe no vetor de saída. A atribuição do valor 0 significa que a classe não foi atribuída ao exemplo.

A escolha de um valor ótimo para o limiar não é uma tarefa trivial. A escolha de um valor baixo para o limiar faz com que muitas classes sejam atribuídas a cada exemplo, resultando em um alto valor de revocação e um baixo valor de precisão. A escolha de um valor de limiar alto, por outro lado, resulta em muito poucas classes atribuídas a cada exemplo, o que faz com que seja obtido um valor alto de precisão e um valor baixo de revocação. Para evitar o problema da

escolha de um limiar adequado, foram utilizadas curvas precisão-revoação (curvas **PR**) como medida de avaliação para comparar os diferentes métodos. Para se obter uma curva **PR** para um dado método de classificação, diferentes valores de limiares variando num intervalo $[0, 1]$ são aplicados às saídas dos métodos. Assim, diferentes valores de precisão e revocação são obtidos, um para cada valor de limiar. Cada valor de precisão e revocação então representa um ponto no espaço **PR**. A união desses pontos forma uma curva **PR**, e a área abaixo dessa curva pode ser calculada. Diferentes métodos podem então ser comparados com base nas áreas abaixo de suas respectivas curvas **PR**.

Para calcular a área abaixo de uma curva **PR**, inicialmente os pontos obtidos com a utilização dos limiares devem ser interpolados (Davis e Goadrich, 2006). Essa interpolação garante que a área abaixo da curva não seja aumentada artificialmente, o que pode acontecer se as curvas forem construídas apenas conectando os pontos sem interpolação. Nesta pesquisa, as curvas **PR** foram utilizadas de duas maneiras. Os métodos foram comparados utilizando a área abaixo da curva **PR** média ($AU(\overline{PRC})$), e a média ponderada das áreas abaixo das curvas **PR** individuais (por classe) ($\overline{AUPRC}_w$).

Para verificar a significância estatística dos resultados, foram empregados os testes de Friedman e Nemenyi, recomendados para comparações envolvendo vários métodos e muitos conjuntos de dados (Demšar, 2006). O teste de Friedman é um teste não paramétrico, pois não assume nada acerca da distribuição dos dados. Ele é baseado em *rankings*, e verifica se há diferença estatisticamente significativa em um grupo de comparações. O teste de Nemenyi é um teste *post-hoc* aplicado após o teste de Friedman. Ele é utilizado para encontrar quais pares de comparações apresentam diferenças estatisticamente significantes. Nos testes estatísticos, foi utilizado um nível de confiança de 95%. As próximas subseções apresentam o cálculo das medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Nas equações apresentadas, i varia de 1 até o número total de classes $|C|$ do problema.

Área abaixo da curva **PR** média

Dado um valor de limiar, um ponto precisão-revoação ($\overline{Prec}, \overline{Rec}$) no espaço **PR** é obtido por meio das Equações 6.1 e 6.2. Essas equações correspondem às micro médias da precisão e da revocação, e assim como as medidas de Precisão e Revocação Hierárquicas propostas em (Kiritchenko et al., 2005), consideram os relacionamentos hierárquicos entre as classes, ou seja, consideram que um exemplo pertence não apenas a uma classe, mas também a todas as classes ancestrais dessa classe na estrutura hierárquica. Nessas equações, i varia de 1 até o número total de classes $|C|$ do problema.

$$\overline{Prec} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FP_i} \quad (6.1)$$

$$\overline{Rec} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FN_i} \quad (6.2)$$

Média ponderada das áreas abaixo das curvas PR individuais

Para calcular a média ponderada das áreas abaixo das curvas PR individuais, primeiro são calculadas as áreas abaixo das curvas PR individuais obtidas em cada classe do problema ($AUPRC_i$). A média ponderada dessas áreas é então obtida por meio da Equação 6.3.

$$\overline{AUPRC}_w = \sum_i w_i \cdot AUPRC_i \quad (6.3)$$

Na Equação (6.3), w_i é utilizado para ponderar a contribuição de cada classe de acordo sua frequência, ou seja, $w_i = v_i / \sum_j v_j$, sendo v_i a frequência da classe c_i no conjunto de dados (Vens et al., 2008). A ideia aqui é que as classes mais frequentes sejam consideradas mais importantes.

Além das curvas PR terem sido escolhidas pelo fato de permitirem uma avaliação independente de limiares, elas foram escolhidas também porque as Equações 6.1 e 6.2 podem ser aplicadas diretamente a qualquer tipo de hierarquia, tanto estruturada como árvore ou grafo, e também podem ser utilizadas diretamente na avaliação de qualquer tipo de classificador, tanto local ou global, e que utilize qualquer tipo de estratégia de classificação, obrigatória ou não obrigatória em nós-folha (Silla e Freitas, 2010). Além disso, experimentos preliminares desta tese mostraram que não existe nenhuma desvantagem na utilização das medidas de precisão e revocação, e que essas medidas são consistentes com outras medidas da literatura, possuindo a vantagem de não dependerem de nenhum parâmetro fornecido por usuários (Cerri et al., 2013).

6.2 Métodos da Literatura Utilizados para Comparação

- **Clus-HMC:** método baseado na abordagem de classificação global, utilizando árvores de decisão. As árvores são construídas utilizando o conceito de *Predictive Clustering Trees* (PCT) (Blockeel et al., 1998), no qual árvores de decisão são vistas como uma hierarquia de grupos (*clusters*). Ao final do processo de indução do método, uma única árvore de decisão é produzida. Atualmente é considerado o método de classificação estado-da-arte na literatura;
- **Clus-HSC:** variação local do método Clus-HMC, também baseada no conceito de PCT. Nesse método, uma árvore de decisão é construída para cada classe da hierarquia, levando em consideração os relacionamentos hierárquicos entre as classes. Isso significa que, para uma dada classe, uma árvore de decisão é induzida utilizando os exemplos atribuídos à classe pai dessa classe;
- **Clus-SC:** método local também baseado no conceito de PCT. Nesse método, uma árvore de decisão é induzida para cada classe da hierarquia, sem considerar os relacionamentos hierárquicos entre as classes;
- **hmAnt-Miner:** método baseado na abordagem global, utilizando Otimização por Colônias de Formigas (OCF). Nesse método, regras de classificação globais são construídas,

ou seja, regras que sejam capazes de atribuir um conjunto de classes a um exemplo, respeitando as restrições da hierarquia de classes. As regras são construídas por uma colônia de formigas, por meio de um procedimento de cobertura sequencial de exemplos.

Além do fato de o método Clus-HMC ser considerado o estado-da-arte na literatura, todos os métodos foram escolhidos para os experimentos também porque estão disponíveis para *download* e execução. Além disso, possuem o mesmo tipo de saída dos métodos HMC-LMLP e HMC-GA, estruturada como um vetor de classes para cada exemplo, no qual cada posição j corresponde à pertinência do exemplo à classe c_j . Ainda, eles foram aplicados aos mesmos conjuntos de dados utilizados nesta Tese.

6.3 Conjuntos de Dados

Nos experimentos, foram utilizados conjuntos de dados¹ cujas classes são estruturados hierarquicamente como árvores e grafos. Os dados são relacionados à tarefa de predição de funções de proteínas, e as taxonomias hierárquicas nas quais as classes estão estruturadas são organizadas de acordo com dois esquemas muito utilizados na literatura, o esquema FunCat (Ruepp et al., 2004) e a *Gene Ontology* (GO) (Ashburner et al., 2000). Os atributos nos conjuntos de dados estruturados como árvores e grafos são os mesmos. A única mudança é o esquema de organização das classes. Esses dois esquemas são descritos a seguir.

O esquema FunCat é um dos mais populares esquemas de organização funcional de proteínas de origem eucariótica e procariótica. Sua hierarquia é organizada em uma árvore com até seis níveis de profundidade, possuindo 28 categorias principais (classes do primeiro nível hierárquico) relacionadas a funções protéicas como transporte celular, metabolismo, comunicação celular e ciclo celular (Ruepp et al., 2004). A Figura 6.1 ilustra parte do esquema FunCat. Como pode ser observado, uma classe é separada de sua subclasse pelo símbolo “.”. Dessa forma, a classe *Amino acid metabolism* (01.01) é uma subclasse da classe *Metabolism* (01).

A Tabela 6.1 apresenta as principais características dos conjuntos de dados utilizados nos experimentos envolvendo hierarquias estruturadas como árvores. Outras características desses dados são apresentadas nas Tabelas 6.2 e 6.3. Por meio dos valores apresentados na Tabela 6.2, é possível observar que muito poucos exemplos possuem classes nos últimos níveis hierárquicos, o que torna a tarefa de classificação muito difícil nesses níveis.

Um segundo esquema de organização hierárquica utilizado foi a *Gene Ontology*, que organiza as classes em uma ontologia de termos, na qual cada termo representa uma função protéica. A taxonomia da GO é organizada como um grafo acíclico direcionado (*Directed Acyclic Graph* (DAG)). Trata-se de um conjunto de três ontologias, cada uma cobrindo um domínio diferente. Os domínios cobertos são componentes celulares, processos biológicos e funções moleculares (Ashburner et al., 2000), possuindo milhares de classes. As hierarquias dos conjuntos de dados utilizados possuem mais de 4 mil classes. A Figura 6.2 ilustra um exemplo de parte da taxonomia de termos da GO.

¹<http://www.cs.kuleuven.be/~dtai/clus/hmcdatasets.html>

Figura 6.1: Parte do esquema de classificação FunCat. Adaptado de <http://mips.helmholtz-muenchen.de/proj/funcatDB>.

Tabela 6.1: Características dos conjuntos de dados estruturados como árvores. A tabela mostra o número de atributos ($|A|$), o número de classes ($|C|$), o número total de exemplos (Total) e o número total de exemplos multirrótulo (Multirrótulo).

Conjunto de dados	$ A $	$ C $	Treinamento		Validação		Teste	
			Total	Multirrótulo	Total	Multirrótulo	Total	Multirrótulo
Cellcycle	77	499	1628	1323	848	673	1281	1059
Church	27	499	1630	1322	844	670	1281	1057
Derisi	63	499	1608	1309	842	671	1275	1055
Eisen	79	461	1058	900	529	441	837	719
Expr	551	499	1639	1328	849	674	1291	1064
Gasch1	173	499	1634	1325	846	672	1284	1059
Gasch2	52	499	1639	1328	849	674	1291	1064
Pheno	69	455	656	537	353	283	582	480
Seq	478	499	1701	1344	879	679	1339	1079
Spo	80	499	1600	1301	837	666	1266	1047

Tabela 6.2: Estatísticas dos conjuntos de dados estruturados como árvores. A tabela mostra, em cada nível, o número médio de exemplos por classe e o número médio de classes por exemplo.

Conjunto de dados	Número médio de exemplos por classe						Número médio de classes por exemplo					
	N1	N2	N3	N4	N5	N6	N1	N2	N3	N4	N5	N6
Treinamento												
Celcycle	90.44	20.12	8.26	6.86	3.93	2.75	2.40	2.84	2.18	1.06	0.22	0.006
Church	90.55	20.15	8.28	6.87	3.92	2.75	2.40	2.83	2.18	1.05	0.22	0.006
Derisi	89.33	19.87	8.17	6.82	3.89	2.75	2.41	2.85	2.19	1.06	0.22	0.006
Eisen	58.77	13.86	6.04	5.09	3.13	2.00	2.48	2.98	2.35	1.13	0.24	0.007
Expr	91.05	20.26	8.32	6.89	3.93	2.75	2.39	2.83	2.17	1.05	0.22	0.006
Gasch1	90.77	20.20	8.29	6.88	3.93	2.75	2.39	2.83	2.17	1.05	0.22	0.006
Gasch2	91.05	20.26	8.32	6.89	3.93	2.75	2.39	2.83	2.17	1.05	0.22	0.006
Pheno	36.44	8.77	3.59	3.12	2.04	1.50	2.54	3.01	2.29	1.08	0.23	0.009
Seq	94.50	20.48	8.42	6.97	3.96	2.75	2.37	2.78	2.13	1.03	0.21	0.006
Spo	88.88	19.77	8.13	6.78	3.88	2.75	2.40	2.84	2.19	1.06	0.22	0.006
Validação												
Celcycle	47.11	10.45	4.24	3.46	2.12	2.50	2.39	2.79	2.13	1.02	0.22	0.011
Church	46.88	10.40	4.22	3.45	2.12	2.50	2.39	2.79	2.13	1.02	0.22	0.011
Derisi	46.77	10.38	4.21	3.44	2.12	2.50	2.40	2.80	2.14	1.01	0.23	0.011
Eisen	29.38	6.90	2.98	2.46	1.55	1.75	2.47	2.93	2.28	1.10	0.23	0.013
Expr	47.16	10.46	4.25	3.47	2.12	2.50	2.39	2.79	2.13	1.02	0.22	0.011
Gasch1	47.00	10.42	4.23	3.45	2.12	2.50	2.39	2.79	2.13	1.02	0.22	0.011
Gasch2	47.16	10.46	4.25	3.47	2.12	2.50	2.39	2.79	2.13	1.02	0.22	0.011
Pheno	19.61	4.71	1.91	1.58	1.06	1.50	2.49	2.87	2.13	1.02	0.22	0.016
Seq	48.83	10.52	4.28	3.50	2.12	2.50	2.35	2.71	2.07	0.99	0.22	0.011
Spo	46.50	10.32	4.19	3.42	2.11	2.50	2.39	2.80	2.14	1.02	0.23	0.011
Teste												
Celcycle	71.16	15.90	6.53	5.39	3.18	2.00	2.46	2.92	2.23	1.05	0.22	0.006
Church	71.16	15.90	6.53	5.38	3.15	2.00	2.46	2.91	2.23	1.05	0.22	0.006
Derisi	70.83	15.82	6.47	5.35	3.15	2.00	2.47	2.93	2.23	1.05	0.22	0.006
Eisen	46.50	10.97	4.75	3.94	2.40	1.00	2.54	3.03	2.36	1.10	0.23	0.004
Expr	71.72	16.02	6.58	5.40	3.18	2.00	2.46	2.91	2.23	1.04	0.22	0.006
Gasch1	71.33	15.93	6.55	5.38	3.15	2.00	2.46	2.91	2.22	1.04	0.22	0.006
Gasch2	71.72	16.02	6.58	5.40	3.18	2.00	2.46	2.91	2.23	1.04	0.22	0.006
Pheno	32.33	7.81	3.17	2.69	1.66	0.75	2.58	3.03	2.24	1.06	0.22	0.005
Seq	74.38	16.22	6.67	5.45	3.22	2.00	2.42	2.83	2.17	1.02	0.21	0.005
Spo	70.33	15.71	6.43	5.33	3.15	2.00	2.47	2.92	2.23	1.05	0.22	0.006

Tabela 6.3: Número de classes por nível nos conjuntos de dados estruturados como árvores

Conjunto de dados	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Celcycle	18	80	178	142	77	4
Church	18	80	178	142	77	4
Derisi	18	80	178	142	77	4
Eisen	18	76	165	131	67	4
Expr	18	80	178	142	77	4
Gasch1	18	80	178	142	77	4
Gasch2	18	80	178	142	77	4
Pheno	18	74	165	129	65	4
Seq	18	80	178	142	77	4
Spo	18	80	178	142	77	4

Figura 6.2: Parte do esquema de classificação da Gene Ontology. Adaptado de <http://www.geneontology.org/>.

Como pode ser observado na Figura 6.2, a principal característica das hierarquias estruturadas como DAGs é que uma classe pode possuir mais de uma superclasse. Isso significa que existe mais de um caminho entre o nó raiz e uma dada classe da hierarquia. Como consequência, a profundidade de uma classe na hierarquia pode variar dependendo do caminho escolhido entre o nó raiz e a classe. Dessa forma, o nível em que está localizada uma classe não pode ser diretamente definido como nas hierarquias estruturadas como árvores. Para definir o nível de uma classe, é necessário que algum critério seja utilizado. Por exemplo, o nível de uma classe pode ser dado pelo número de arestas localizadas no menor caminho entre o nó raiz e a classe, ou então pelo número de arestas do maior caminho. A Tabela 6.4 apresenta algumas características dos conjuntos de dados estruturados como DAGs.

Como pode ser observado na Tabela 6.4, diferentemente dos conjuntos de dados estruturados como árvores, todos os exemplos dos conjuntos de dados estruturados como grafos são exemplos multirrótulo. Não foram computadas estatísticas por nível justamente porque não há definição clara dos níveis hierárquicos das classes na Gene Ontology. Nos experimentos realizados com o método HMC-LMLP (Seção 6.4), as hierarquias estruturadas como DAGs precisaram ser adaptadas para o treinamento de uma rede neural por nível hierárquico. Assim, estatísticas foram computadas para cada nível, e serão apresentadas na Seção 6.4. Já nos experimentos como o método HMC-GA (Seção 6.5), por esse se tratar de um método global, foram utilizados os conjuntos de dados originais, já que nenhuma definição explícita dos níveis hierárquicos das classes é necessária.

Tabela 6.4: Características dos conjuntos de dados estruturados como grafos. A tabela mostra o número de atributos ($|A|$), o número de classes ($|C|$), o número total de exemplos (Total) e o número total de exemplos multirrótulo (Multirrótulo).

Conjunto de dados	$ A $	$ C $	Treinamento		Validação		Teste	
			Total	Multirrótulo	Total	Multirrótulo	Total	Multirrótulo
Celleycle	77	4122	1625	1625	848	848	1278	1278
Church	27	4122	1627	1627	844	844	1278	1278
Derisi	63	4116	1605	1605	842	842	1272	1272
Eisen	79	3570	1055	1055	528	528	835	835
Expr	551	4128	1636	1636	849	849	1288	1288
Gasch1	173	4122	1631	1631	846	846	1281	1281
Gasch2	52	4128	1636	1636	849	849	1288	1288
Pheno	69	3124	653	653	352	352	581	581
Seq	478	4130	1692	1692	876	876	1332	1332
Spo	80	4116	1597	1597	837	837	1263	1263

Ainda nos conjuntos de dados estruturados como grafos, durante a fase de avaliação dos métodos, não foram consideradas as classes GO0003674, GO0005575, GO0008150. Essas classes foram deixadas fora da avaliação por se tratarem das classes raízes das três hierarquias que compõem a Gene Ontology, respectivamente funções moleculares, componentes celulares e processos biológicos. Sendo assim, todos os exemplos dos conjuntos de dados são classificados nessas classes.

Em todos os experimentos, envolvendo ambos os conjuntos de dados estruturados como árvores ou grafos, as partições de treino, validação e teste são exatamente as mesmas originalmente utilizadas em (Vens et al., 2008). Assim, os classificadores são induzidos utilizando os conjuntos de treinamento e validação, e a avaliação final é realizada utilizando os conjuntos de teste. Optou-se pela utilização das três partições fixas de dados porque todos os trabalhos da literatura que utilizaram esses dados os utilizaram exatamente dessa maneira, com as mesmas partições disponibilizados por Vens et al. (2008). Isso facilitou a comparação dos métodos propostos com os métodos da literatura, e permitiu que os resultados da literatura fossem reproduzidos.

Nas próximas seções são apresentados os experimentos realizados com os métodos HMC-LMLP e HMC-GA. Nas tabelas mostrando os resultados dos experimentos, observa-se que os valores de $AU(\overline{PRC})$ e $\overline{AUPRC}_w$ apresentados são bem baixos. No entanto, deve ser considerada a dificuldade do problema de classificação, que possui centenas ou milhares de classes. Assim, os valores apresentados são comuns considerando a área de pesquisa.

6.4 Experimentos com o Método HMC-LMLP

Nesta seção são apresentados os experimentos realizados utilizando o método HMC-LMLP. Como explicado no Capítulo 4, o método utiliza uma rede neural MLP associada a cada nível hierárquico. Assim, é necessária a definição clara dos níveis a que pertencem cada classe. Dessa forma, foi feita uma adaptação dos conjuntos de dados estruturados como grafos para que os mesmos sejam utilizados no método HMC-LMLP. A adaptação consistiu em definir o nível de uma classe como o número de arestas pertencentes ao caminho mais longo entre o nó raiz

e a classe. O caminho mais longo foi escolhido porque assim há a garantia de que quando uma classe estiver localizada no nível l , todas as suas superclasses estarão localizadas em níveis anteriores a l , ou seja, níveis mais próximos à raiz do que o nível l . As Tabelas 6.5, 6.6 e 6.7 apresentam algumas características dos conjuntos de dados estruturados como DAGs, após o procedimento de adaptação que definiu os níveis das classes.

Tabela 6.5: Número de classes por nível nos conjuntos de dados estruturados como grafos

Dados	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Cellcycle	33	155	394	597	929	779	631	335	171	63	21	5	9
Church	33	155	394	597	929	779	631	335	171	63	21	5	9
Derisi	33	155	394	596	927	778	630	334	171	63	21	5	9
Eisen	33	149	360	524	786	679	539	271	141	55	19	5	9
Expr	33	155	394	599	932	780	631	335	171	63	21	5	9
Gasch1	33	155	394	597	929	779	631	335	171	63	21	5	9
Gasch2	33	155	394	599	932	780	631	335	171	63	21	5	9
Pheno	33	145	332	489	670	568	460	236	114	49	18	4	6
Seq	33	155	394	599	932	780	633	335	171	63	21	5	9
Spo	33	155	394	596	927	778	630	334	171	63	21	5	9

Todas as quatro variações do método HMC-LMLP (HMC-LMLP-Labels, HMC-LMLP-Predicted, HMC-LMLP-True e HMC-LMLP-NoLabels) foram executadas utilizando dois diferentes algoritmos de treinamento para redes MLP: o algoritmo *Back-propagation* tradicional (Bp) (Rumelhart e McClelland, 1986) e o algoritmo *Resilient Back-propagation* (Rprop). Como discutido no Capítulo 3, o algoritmo *Resilient Back-propagation* tenta eliminar a influência do tamanho da derivada parcial na mudança dos pesos sinápticos, considerando apenas o sinal da derivada para indicar a direção da atualização de um peso.

6.4.1 Parâmetros

Os parâmetros utilizados nas redes MLP foram otimizados usando os conjuntos de validação. Os parâmetros empregados são listados a seguir.

- número de neurônios em cada camada escondida, iniciando com a camada escondida da MLP associada ao primeiro nível hierárquico, e finalizando com a camada escondida da MLP associada ao último nível hierárquico;
- taxa de aprendizado e constante *momentum* utilizadas no algoritmo *Back-propagation*;
- faixa de valores utilizada para iniciar os pesos das redes neurais;
- valores dos parâmetros do algoritmo *Resilient Back-propagation*.

O número de neurônios de cada camada escondida foi escolhido como sendo uma fração do número de atributos de entrada. Exemplificando, o valor 0.5 significa que o número de neurônios da camada escondida é igual a 50% do número de atributos de entrada. Para o algoritmo Rprop, os valores utilizados nos parâmetros foram sugeridos em (Riedmiller e Braun, 1993), e nenhuma tentativa de otimização foi realizada.

Tabela 6.6: Número médio de exemplos por classe nos conjuntos de dados estruturados como grafos

Conjunto de dados	Número médio de exemplos por classe												
	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13
Treinamento													
Celcycle	48.42	10.31	4.04	2.63	1.65	1.68	1.70	1.96	2.38	3.06	3.52	3.60	1.11
Church	48.48	10.32	4.04	2.63	1.65	1.68	1.70	1.95	2.37	3.06	3.52	3.60	1.11
Derisi	47.82	10.18	3.99	2.60	1.63	1.66	1.69	1.94	2.36	3.05	3.52	3.60	1.11
Eisen	31.94	7.07	2.92	2.00	1.31	1.34	1.42	1.76	2.16	2.62	3.11	3.20	1.11
Expr	48.76	10.38	4.07	2.64	1.65	1.68	1.70	1.96	2.38	3.06	3.52	3.60	1.11
Gasch1	48.61	10.35	4.05	2.64	1.65	1.68	1.70	1.96	2.37	3.06	3.52	3.60	1.11
Gasch2	48.76	10.38	4.07	2.64	1.65	1.68	1.70	1.96	2.38	3.06	3.52	3.60	1.11
Pheno	19.36	4.41	1.92	1.29	0.91	0.90	0.91	1.09	1.40	1.49	1.39	2.00	1.00
Seq	50.45	10.74	4.21	2.73	1.71	1.70	1.71	1.98	2.42	3.16	3.52	3.60	1.11
Spo	47.58	10.13	3.97	2.59	1.62	1.65	1.68	1.94	2.35	3.03	3.52	3.60	1.11
Validação													
Celcycle	24.97	5.32	2.09	1.36	0.85	0.86	0.85	0.95	1.10	1.48	1.76	1.40	0.44
Church	24.85	5.29	2.08	1.35	0.85	0.86	0.85	0.95	1.09	1.46	1.71	1.20	0.33
Derisi	24.79	5.28	2.07	1.36	0.85	0.86	0.85	0.96	1.11	1.48	1.76	1.40	0.33
Eisen	15.97	3.54	1.46	1.00	0.66	0.67	0.69	0.85	1.00	1.16	1.58	0.80	0.22
Expr	25.00	5.32	2.09	1.36	0.85	0.86	0.85	0.95	1.11	1.48	1.76	1.40	0.44
Gasch1	24.91	5.30	2.08	1.36	0.85	0.86	0.85	0.95	1.09	1.46	1.71	1.20	0.33
Gasch2	25.00	5.32	2.09	1.36	0.85	0.86	0.85	0.95	1.11	1.48	1.76	1.40	0.44
Pheno	10.42	2.37	1.03	0.70	0.50	0.49	0.48	0.59	0.72	0.88	0.83	0.50	0.17
Seq	25.82	5.50	2.16	1.40	0.88	0.87	0.85	0.96	1.12	1.49	1.76	1.40	0.44
Spo	24.64	5.25	2.06	1.35	0.85	0.85	0.84	0.94	1.09	1.46	1.76	1.40	0.33
Teste													
Celcycle	38.12	8.12	3.19	2.08	1.31	1.34	1.36	1.49	1.87	2.49	2.95	2.60	0.78
Church	38.12	8.12	3.18	2.08	1.30	1.34	1.36	1.50	1.87	2.49	2.95	2.60	0.78
Derisi	37.94	8.08	3.17	2.08	1.30	1.34	1.36	1.50	1.88	2.49	2.95	2.60	0.78
Eisen	25.27	5.60	2.32	1.59	1.04	1.09	1.14	1.32	1.61	2.13	2.32	2.20	0.78
Expr	38.42	8.18	3.21	2.09	1.31	1.35	1.37	1.50	1.87	2.49	2.95	2.60	0.78
Gasch1	38.21	8.14	3.19	2.08	1.31	1.35	1.36	1.50	1.87	2.49	2.95	2.60	0.78
Gasch2	38.42	8.18	3.21	2.09	1.31	1.35	1.37	1.50	1.87	2.49	2.95	2.60	0.78
Pheno	17.36	3.95	1.72	1.15	0.82	0.81	0.83	0.97	1.22	1.49	1.83	2.25	0.83
Seq	39.73	8.46	3.32	2.16	1.36	1.37	1.38	1.51	1.88	2.51	2.95	2.60	0.78
Spo	37.67	8.02	3.15	2.06	1.29	1.33	1.35	1.49	1.87	2.49	2.95	2.60	0.78

Tabela 6.7: Número médio de classes por exemplo nos conjuntos de dados estruturados como grafos

Conjunto de dados	Número médio de classes por exemplo												
	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13
Treinamento													
Cellcycle	5.23	6.64	6.35	4.70	3.75	2.29	1.47	0.67	0.34	0.15	0.06	0.01	0.01
Church	5.23	6.63	6.35	4.70	3.75	2.29	1.47	0.67	0.34	0.15	0.06	0.01	0.01
Derisi	5.24	6.65	6.37	4.71	3.77	2.30	1.48	0.67	0.34	0.15	0.06	0.01	0.01
Eisen	5.69	7.25	6.93	5.13	4.18	2.61	1.68	0.79	0.40	0.17	0.07	0.02	0.01
Expr	5.23	6.63	6.34	4.69	3.75	2.29	1.46	0.67	0.34	0.15	0.06	0.01	0.01
Gasch1	5.23	6.63	6.35	4.70	3.75	2.29	1.46	0.67	0.34	0.15	0.06	0.01	0.01
Gasch2	5.23	6.63	6.34	4.69	3.75	2.29	1.46	0.67	0.34	0.15	0.06	0.01	0.01
Pheno	5.22	6.70	6.47	4.79	3.75	2.33	1.49	0.64	0.33	0.14	0.05	0.02	0.01
Seq	5.22	6.60	6.32	4.64	3.73	2.25	1.43	0.65	0.33	0.15	0.05	0.01	0.01
Spo	5.24	6.65	6.36	4.71	3.76	2.30	1.48	0.68	0.34	0.15	0.06	0.02	0.01
Validação													
Cellcycle	5.13	6.51	6.22	4.59	3.65	2.09	1.35	0.63	0.32	0.15	0.06	0.01	0.01
Church	5.13	6.51	6.22	4.58	3.66	2.09	1.35	0.63	0.32	0.14	0.06	0.009	0.005
Derisi	5.15	6.53	6.25	4.61	3.67	2.10	1.35	0.64	0.32	0.15	0.06	0.01	0.005
Eisen	5.64	7.19	6.82	5.04	4.06	2.38	1.53	0.73	0.38	0.17	0.08	0.01	0.01
Expr	5.13	6.51	6.22	4.58	3.65	2.09	1.35	0.63	0.32	0.14	0.06	0.01	0.01
Gasch1	5.13	6.51	6.22	4.58	3.65	2.09	1.35	0.63	0.32	0.14	0.06	0.009	0.005
Gasch2	5.13	6.51	6.22	4.58	3.65	2.09	1.35	0.63	0.32	0.14	0.06	0.01	0.01
Pheno	5.27	6.70	6.54	4.79	3.76	2.15	1.45	0.71	0.33	0.15	0.06	0.01	0.01
Seq	5.11	6.46	6.17	4.51	3.62	2.04	1.32	0.62	0.31	0.14	0.06	0.01	0.01
Spo	5.15	6.52	6.24	4.61	3.67	2.09	1.35	0.64	0.32	0.15	0.06	0.01	0.005
Teste													
Cellcycle	5.42	6.89	6.63	4.92	3.94	2.38	1.54	0.69	0.35	0.16	0.06	0.01	0.01
Church	5.43	6.88	6.63	4.91	3.93	2.37	1.54	0.70	0.35	0.16	0.06	0.01	0.01
Derisi	5.42	6.88	6.65	4.93	3.94	2.39	1.54	0.70	0.35	0.16	0.06	0.01	0.01
Eisen	5.82	7.43	7.15	5.34	4.29	2.65	1.74	0.76	0.38	0.18	0.07	0.02	0.01
Expr	5.43	6.89	6.62	4.91	3.93	2.37	1.53	0.69	0.35	0.16	0.06	0.01	0.01
Gasch1	5.43	6.89	6.63	4.91	3.93	2.37	1.53	0.69	0.35	0.16	0.06	0.01	0.01
Gasch2	5.43	6.89	6.62	4.91	3.93	2.37	1.53	0.69	0.35	0.16	0.06	0.01	0.01
Pheno	5.38	6.88	6.71	5.02	3.90	2.41	1.62	0.74	0.36	0.17	0.08	0.02	0.01
Seq	5.41	6.84	6.57	4.84	3.90	2.32	1.49	0.67	0.34	0.15	0.06	0.01	0.01
Spo	5.42	6.88	6.64	4.92	3.94	2.38	1.54	0.70	0.35	0.16	0.06	0.02	0.01

Como os conjuntos de dados estruturados como árvores e grafos possuem os mesmos valores de atributos (apenas a estrutura hierárquica é diferente), e também devido ao alto custo computacional da execução do método HMC-LMLP nos conjuntos de dados estruturados como grafos, a otimização de parâmetros foi realizada considerando apenas os conjuntos de dados estruturados como árvores. Para a otimização dos parâmetros, o método HMC-LMLP-Predicted foi executado utilizando diferentes conjuntos de valores para os parâmetros. Foram utilizados diferentes valores de pesos iniciais, números de neurônios das camadas escondidas, valores de taxa de aprendizado e valores de constante de momentum. Não foram utilizados todos os possíveis valores porque o número de possibilidades é muito grande, e também porque a otimização de parâmetros não foi o foco da pesquisa.

Para os valores iniciais dos pesos sinápticos, notou-se que quanto maior seus valores, mais suscetível a *overfitting* as redes neurais se tornavam, obtendo melhor desempenho nas classes mais frequentes, porém um pior desempenho geral. Os valores dos pesos foram inicialmente selecionados aleatoriamente no intervalo $[-0.1, 0.1]$, e então esse intervalo foi sendo gradualmente aumentado até o intervalo $[-1, 1]$. Para o número de neurônios das camadas escondidas, um número limitado de valores foi testado. Inicialmente foram utilizados os valores 1.0/0.9/0.8/0.7/0.6/0.5 para cada camada escondida em cada nível hierárquico. Esses valores foram então gradualmente reduzidos até os valores 0.1/0.08/0.06/0.04/0.03/0.02. Como mencionado anteriormente, esses valores representam frações do número total de entradas das redes neurais. Assim, se uma rede neural possui 100 entradas, o valor 0.6 significa que ela possui 60 neurônios em sua camada escondida. Considerando a taxa de aprendizado e constante *momentum*, inicialmente foram utilizados os mesmos valores usados na plataforma Weka (Hall et al., 2009), que são 0.3 para a taxa de aprendizado e 0.2 para a constante *momentum*. Esses valores foram gradualmente reduzidos e observou-se que as redes neurais tornavam-se menos suscetíveis a *overfitting* conforme esses valores eram diminuídos. Os valores finais obtidos para os parâmetros são listados a seguir.

- Número de neurônios das camadas escondidas, por nível:
 - Árvore: 0.6/0.5/0.4/0.3/0.2/0.1;
 - Grafo: 0.65/0.65/0.6/0.55/0.5/0.45/0.4/0.35/0.3/0.25/0.2/0.15/0.1;
- Taxa de aprendizado e constante *momentum* utilizadas no algoritmo *Back-propagation* nas camadas escondida e de saída, respectivamente: {0.05, 0.03} e {0.02, 0.01};
- Pesos iniciais das redes neurais gerados aleatoriamente no intervalo $[-0.1, 0.1]$;
- Valores dos parâmetros do algoritmo *Resilient Back-propagation*: Delta inicial (Δ_0) = 0.1, Delta máximo (Δ_{max}) = 50.0, Delta mínimo (Δ_{min}) = $1e^{-6}$, fator de aumento (η^+) = 1.2 e fator de decréscimo (η^-) = 0.5.

Observa-se que o número de neurônios das camadas escondidas é reduzido conforme o nível hierárquico torna-se mais profundo. Isso foi feito para tentar amenizar o *overfitting* das redes

neurais, já que o número de exemplos de treinamento torna-se menor a medida que a profundidade do nível hierárquico aumenta. Ainda, a fração do número de neurônios em cada camada escondida nas hierarquias estruturadas como grafos foi escolhida de maneira que os valores ficassem próximos aos valores utilizados nas árvores. Os dois primeiros valores (0.65/0.65) são iguais porque todos os exemplos são atribuídos a classes do primeiro e segundo níveis. A seguir são apresentados os resultados dos experimentos realizados.

6.4.2 Experimentos

Experimentos comparando as diferentes variações do método HMC-LMLP

As tabelas 6.8 e 6.9 apresentam os resultados obtidos com as quatro variações do método HMC-LMLP, utilizando os algoritmos de treinamento *Back-propagation* (Bp-Labels, Bp-Predicted, Bp-True e Bp-NoLabels) e *Resilient Back-propagation* (Rp-Labels, Rp-Predicted, Rp-True, Rp-NoLabels). Os melhores resultados são destacados em negrito.

Como pode ser observado na Tabela 6.8, a variação Bp-Predicted obteve os melhores resultados na maioria dos conjuntos de dados cujas classes são estruturadas como árvores. Isso reforça a tese de que a utilização das predições para complementar os vetores de atributos dos exemplos leva à melhora do desempenho de classificação.

Ainda nas hierarquias estruturadas como árvores, além da utilização das predições (Bp-Predicted e Rp-Predicted) proporcionar a melhora do desempenho preditivo se comparado com as variações que utilizam apenas os atributos originais dos exemplos (Bp-NoLabels e Rp-NoLabels), observa-se uma melhora de desempenho mais acentuada quando as variações Bp-Predicted e Rp-Predicted são comparadas com as variações que utilizam as classes verdadeiras para complementar os vetores de atributos dos exemplos durante o treinamento (Bp-True e Rp-True). Isso contribui para a tese de que as redes neurais foram capazes elas mesmas de aprender os relacionamentos entre as classes da hierarquia, além de tornar possível que informações previamente aprendidas por uma rede neural sejam utilizadas pela rede neural localizada no próximo nível. O uso das classes verdadeiras parece ter prejudicado o aprendizado também pelo fato de diferentes informações serem utilizadas no treinamento e teste das redes neurais. Enquanto as classes verdadeiras são utilizadas no treinamento, as predições devem ser utilizadas no teste. Neste caso, porém, as redes sendo testadas não foram treinadas com informações aprendidas por elas mesmas.

É possível ainda observar na Tabela 6.8 que o método Bp-Predicted obteve os melhores resultados na maioria dos conjuntos de dados considerando as duas variações das curvas PR , $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Isso é indicativo de que a variação Bp-Predicted foi melhor que as outras variações nas classes mais frequentes da hierarquia, consideradas as mais importantes pela medida $\overline{AUPRC}_w$.

Considerando os resultados nos conjuntos de dados estruturados como grafos (Tabela 6.9), pode-se observar um maior equilíbrio entre os resultados obtidos, principalmente considerando a medida $\overline{AUPRC}_w$. Apesar de a variação Rp-NoLabels ter obtido os melhores resultados com a

Tabela 6.8: Resultados das quatro variações do método HMC-LMLP nos conjuntos de dados estruturados como árvores

Dados	Bp-Labels	Bp-Predicted	Bp-True	Bp-NoLabels	Rp-Labels	Rp-Predicted	Rp-True	Rp-NoLabels
$AU(\overline{PRC})$								
Cellcycle	0.185 ± 0.001	0.207 ± 0.001	0.203 ± 0.001	0.205 ± 0.001	0.183 ± 0.002	0.202 ± 0.002	0.199 ± 0.002	0.199 ± 0.002
Church	0.164 ± 0.001	0.173 ± 0.001	0.167 ± 0.002	0.169 ± 0.001	0.163 ± 0.001	0.170 ± 0.001	0.166 ± 0.002	0.168 ± 0.001
Derisi	0.171 ± 0.001	0.183 ± 0.001	0.176 ± 0.001	0.182 ± 0.001	0.170 ± 0.001	0.177 ± 0.001	0.176 ± 0.001	0.177 ± 0.001
Eisen	0.208 ± 0.001	0.245 ± 0.002	0.236 ± 0.001	0.240 ± 0.001	0.206 ± 0.002	0.239 ± 0.002	0.237 ± 0.002	0.238 ± 0.001
Gasch1	0.196 ± 0.001	0.236 ± 0.002	0.229 ± 0.002	0.234 ± 0.002	0.199 ± 0.001	0.235 ± 0.002	0.232 ± 0.003	0.232 ± 0.002
Gasch2	0.184 ± 0.001	0.211 ± 0.001	0.201 ± 0.001	0.208 ± 0.001	0.183 ± 0.003	0.206 ± 0.002	0.200 ± 0.002	0.203 ± 0.002
Pheno	0.159 ± 0.001	0.159 ± 0.002	0.158 ± 0.001	0.159 ± 0.002	0.155 ± 0.002	0.155 ± 0.002	0.147 ± 0.002	0.149 ± 0.003
Spo	0.172 ± 0.001	0.186 ± 0.002	0.180 ± 0.002	0.184 ± 0.001	0.168 ± 0.001	0.178 ± 0.001	0.174 ± 0.002	0.177 ± 0.001
Expr	0.196 ± 0.003	0.243 ± 0.002	0.238 ± 0.003	0.240 ± 0.003	0.200 ± 0.002	0.246 ± 0.004	0.246 ± 0.003	0.244 ± 0.004
Seq	0.195 ± 0.003	0.236 ± 0.002	0.233 ± 0.005	0.232 ± 0.004	0.203 ± 0.002	0.249 ± 0.003	0.248 ± 0.005	0.247 ± 0.002
Média	0.183	0.208	0.202	0.205	0.183	0.206	0.202	0.203
$\overline{AUPRC}_w$								
Cellcycle	0.145 ± 0.001	0.184 ± 0.003	0.178 ± 0.002	0.181 ± 0.002	0.145 ± 0.002	0.181 ± 0.002	0.177 ± 0.002	0.178 ± 0.003
Church	0.118 ± 0.001	0.131 ± 0.002	0.129 ± 0.001	0.127 ± 0.001	0.118 ± 0.002	0.130 ± 0.001	0.128 ± 0.001	0.128 ± 0.001
Derisi	0.127 ± 0.001	0.146 ± 0.002	0.141 ± 0.002	0.144 ± 0.001	0.127 ± 0.001	0.142 ± 0.002	0.140 ± 0.001	0.140 ± 0.002
Eisen	0.163 ± 0.001	0.221 ± 0.003	0.210 ± 0.003	0.213 ± 0.002	0.163 ± 0.003	0.220 ± 0.004	0.214 ± 0.001	0.215 ± 0.003
Gasch1	0.157 ± 0.002	0.213 ± 0.001	0.207 ± 0.003	0.211 ± 0.001	0.158 ± 0.002	0.218 ± 0.002	0.217 ± 0.003	0.216 ± 0.004
Gasch2	0.142 ± 0.001	0.185 ± 0.001	0.174 ± 0.002	0.179 ± 0.003	0.144 ± 0.006	0.180 ± 0.002	0.170 ± 0.002	0.179 ± 0.001
Pheno	0.114 ± 0.001	0.125 ± 0.002	0.118 ± 0.003	0.123 ± 0.002	0.111 ± 0.002	0.120 ± 0.002	0.119 ± 0.001	0.118 ± 0.002
Spo	0.129 ± 0.001	0.152 ± 0.002	0.148 ± 0.002	0.150 ± 0.002	0.125 ± 0.002	0.145 ± 0.002	0.141 ± 0.002	0.144 ± 0.003
Expr	0.167 ± 0.003	0.236 ± 0.002	0.232 ± 0.004	0.233 ± 0.004	0.165 ± 0.002	0.233 ± 0.003	0.234 ± 0.002	0.232 ± 0.004
Seq	0.166 ± 0.002	0.220 ± 0.003	0.218 ± 0.001	0.219 ± 0.002	0.168 ± 0.002	0.240 ± 0.002	0.238 ± 0.003	0.238 ± 0.002
Média	0.143	0.181	0.175	0.178	0.142	0.181	0.178	0.179

Tabela 6.9: Resultados das quatro variações do método HMC-LMLP nos conjuntos de dados estruturados como grafos

Dados	Bp-Labels	Bp-Predicted	Bp-True	Bp-NoLabels	Rp-Labels	Rp-Predicted	Rp-True	Rp-NoLabels
$AU(\overline{PRC})$								
Cellcycle	0.339 ± 0.001	0.361 ± 0.001	0.352 ± 0.002	0.359 ± 0.001	0.344 ± 0.001	0.364 ± 0.001	0.357 ± 0.001	0.365 ± 0.001
Church	0.334 ± 0.001	0.341 ± 0.001	0.336 ± 0.001	0.340 ± 0.001	0.341 ± 0.001	0.347 ± 0.001	0.341 ± 0.003	0.347 ± 0.001
Derisi	0.334 ± 0.001	0.343 ± 0.001	0.336 ± 0.001	0.345 ± 0.001	0.341 ± 0.001	0.344 ± 0.001	0.336 ± 0.002	0.349 ± 0.001
Eisen	0.363 ± 0.002	0.403 ± 0.002	0.393 ± 0.001	0.395 ± 0.001	0.368 ± 0.001	0.404 ± 0.001	0.396 ± 0.003	0.403 ± 0.001
Gasch1	0.343 ± 0.001	0.380 ± 0.002	0.373 ± 0.003	0.378 ± 0.001	0.349 ± 0.001	0.378 ± 0.003	0.372 ± 0.004	0.384 ± 0.001
Gasch2	0.340 ± 0.001	0.369 ± 0.001	0.359 ± 0.002	0.362 ± 0.001	0.347 ± 0.001	0.372 ± 0.001	0.359 ± 0.003	0.369 ± 0.001
Pheno	0.326 ± 0.001	0.318 ± 0.002	0.315 ± 0.002	0.322 ± 0.001	0.333 ± 0.001	0.314 ± 0.002	0.316 ± 0.002	0.325 ± 0.002
Spo	0.333 ± 0.001	0.342 ± 0.001	0.334 ± 0.002	0.340 ± 0.001	0.339 ± 0.001	0.341 ± 0.001	0.332 ± 0.003	0.345 ± 0.001
Expr	0.336 ± 0.002	0.372 ± 0.003	0.369 ± 0.003	0.371 ± 0.001	0.346 ± 0.001	0.377 ± 0.003	0.373 ± 0.003	0.384 ± 0.002
Seq	0.343 ± 0.003	0.370 ± 0.002	0.368 ± 0.003	0.368 ± 0.002	0.352 ± 0.002	0.382 ± 0.002	0.375 ± 0.003	0.384 ± 0.002
Média	0.339	0.360	0.355	0.358	0.346	0.362	0.356	0.365
$\overline{AUPRC}_w$								
Cellcycle	0.242 ± 0.003	0.304 ± 0.002	0.291 ± 0.003	0.295 ± 0.001	0.241 ± 0.002	0.304 ± 0.002	0.297 ± 0.002	0.296 ± 0.002
Church	0.230 ± 0.002	0.257 ± 0.001	0.251 ± 0.002	0.251 ± 0.001	0.231 ± 0.002	0.257 ± 0.002	0.254 ± 0.001	0.252 ± 0.002
Derisi	0.232 ± 0.001	0.267 ± 0.002	0.257 ± 0.001	0.265 ± 0.001	0.236 ± 0.005	0.263 ± 0.002	0.258 ± 0.002	0.260 ± 0.003
Eisen	0.256 ± 0.002	0.345 ± 0.002	0.329 ± 0.002	0.332 ± 0.003	0.258 ± 0.001	0.346 ± 0.003	0.339 ± 0.003	0.335 ± 0.002
Gasch1	0.254 ± 0.005	0.328 ± 0.004	0.321 ± 0.002	0.325 ± 0.002	0.249 ± 0.003	0.329 ± 0.006	0.322 ± 0.003	0.331 ± 0.003
Gasch2	0.246 ± 0.006	0.311 ± 0.001	0.300 ± 0.002	0.297 ± 0.001	0.243 ± 0.002	0.309 ± 0.001	0.300 ± 0.003	0.300 ± 0.002
Pheno	0.223 ± 0.001	0.231 ± 0.001	0.231 ± 0.001	0.229 ± 0.001	0.223 ± 0.001	0.232 ± 0.001	0.232 ± 0.003	0.231 ± 0.001
Spo	0.229 ± 0.002	0.268 ± 0.002	0.257 ± 0.003	0.263 ± 0.001	0.230 ± 0.001	0.261 ± 0.001	0.257 ± 0.003	0.259 ± 0.002
Expr	0.247 ± 0.002	0.332 ± 0.003	0.328 ± 0.003	0.331 ± 0.002	0.254 ± 0.004	0.331 ± 0.002	0.329 ± 0.003	0.335 ± 0.003
Seq	0.260 ± 0.004	0.330 ± 0.003	0.327 ± 0.002	0.328 ± 0.001	0.273 ± 0.003	0.347 ± 0.002	0.341 ± 0.002	0.346 ± 0.003
Média	0.242	0.297	0.289	0.292	0.244	0.298	0.293	0.294

medida $AU(\overline{PRC})$, quando a avaliação considera as frequências das classes, os melhores resultados são obtidos pelas variações Bp-Predicted e Rp-Predicted. Isso é indício de que a variação Rp-NoLabels conseguiu melhores resultados em classes com baixas frequências nos conjuntos de dados, o que diminui o valor da medida $\overline{AUPRC}_w$.

Outro fator que pode ter levado a um maior equilíbrio nos resultados obtidos nas hierarquias estruturadas como grafos foi a adaptação feita nessas hierarquias para a aplicação do método **HMC-LMLP**. Assim, é possível que as variações Bp-Predicted e Rp-Predicted tivessem obtido resultados melhores se todos os relacionamentos entre as classes tivessem sido utilizados durante o processo de treinamento. Como foi feita uma adaptação nas hierarquias para definir a profundidade de uma classe como sendo o número de arestas pertencentes ao caminho mais longo entre a classe e o nó raiz, nem todos os relacionamentos hierárquicos foram considerados durante o treinamento. Exemplificando, considere que um exemplo de treinamento pertence aos caminhos $A.C$ e $A.B.C$ de uma estrutura hierárquica em grafo, e a classe C é uma subclasse direta de ambas as classes A e B , ou seja, há apenas uma aresta conectando A e C , e conectando B e C . Nesse caso, há duas possíveis profundidades para a classe C . Ela pode ter profundidade 2 se for considerado o caminho $A.C$, e profundidade 3 se for considerado o caminho $A.B.C$. Na adaptação realizada, a classe C é definida como pertencendo ao terceiro nível hierárquico. Nesse caso, quando uma rede neural do método **HMC-LMLP** é treinada para o terceiro nível, a classe C é considerada apenas como uma subclasse da classe B . Assim, quando treinando a rede neural para prever a classe C (terceiro nível), não é utilizada como entrada informação relacionada a todas as superclasses da classe C , que seriam as previsões para as classes A e B . Apenas a previsão para a classe B é utilizada.

Comparando a utilização dos algoritmos *Back-propagation* e *Resilient Back-propagation*, observa-se que nas hierarquias estruturadas como árvores, as variações utilizando o algoritmo Rprop obtiveram os melhores desempenhos nos conjuntos de dados Expr e Seq. Já nas hierarquias estruturadas como grafos, as variações utilizando o algoritmo Rprop obtiveram os melhores resultados na maioria dos conjuntos de dados. Esses resultados sugerem que o algoritmo Rprop deve ser utilizado quando os exemplos possuem um número grande de atributos. Os conjuntos de dados Expr e Seq são os que possuem o maior número de atributos, o que pode explicar o melhor desempenho do algoritmo Rprop justamente nesses dois conjuntos de dados em meio a todos os conjuntos de dados estruturados como árvores. Já nas hierarquias estruturadas como grafos, como as mesmas possuem milhares de classes, a utilização das mesmas para complementar os vetores de atributos dos exemplos resulta em um aumento significativo da dimensão dos mesmos. Isso justifica o fato do algoritmo Rprop ter obtido o melhor desempenho na maioria dos conjuntos de dados estruturados como grafos. As Tabelas 6.10 e 6.11 comparam as variações do método **HMC-LMLP** quanto ao número médio de ciclos necessários até a convergência dos algoritmos de treinamento.

Como pode ser observado nas Tabelas 6.10 e 6.11, quando o algoritmo *Back-propagation* foi utilizado, o treinamento terminou prematuramente para os conjuntos de dados que possuem as maiores quantidades de atributos (Gasch1, Expr e Seq). Isso pode ser observado mais clara-

Tabela 6.10: Número médio de ciclos necessários até a convergências dos algoritmos *Back-propagation* e *Resilient Back-propagation* nos conjuntos de dados estruturados como árvores

Dataset	Bp-Labels	Bp-Predicted	Bp-True	Bp-NoLabels
Celcycle	14.47 $\pm$ 3.94	22.20 $\pm$ 3.58	11.10 $\pm$ 1.10	23.80 $\pm$ 3.01
Church	26.67 $\pm$ 6.99	35.60 $\pm$ 6.88	14.90 $\pm$ 2.18	47.00 $\pm$ 10.13
Derisi	18.20 $\pm$ 4.19	26.90 $\pm$ 7.06	11.00 $\pm$ 1.76	28.10 $\pm$ 5.55
Eisen	17.47 $\pm$ 4.37	28.80 $\pm$ 5.12	14.70 $\pm$ 2.87	27.00 $\pm$ 2.79
Gasch1	10.13 $\pm$ 2.33	15.30 $\pm$ 1.49	9.80 $\pm$ 1.55	15.90 $\pm$ 2.68
Gasch2	24.53 $\pm$ 7.56	39.40 $\pm$ 3.24	15.40 $\pm$ 2.17	36.50 $\pm$ 8.49
Pheno	54.47 $\pm$ 21.47	46.60 $\pm$ 17.50	60.50 $\pm$ 17.35	42.30 $\pm$ 11.79
Spo	15.27 $\pm$ 4.23	16.20 $\pm$ 2.90	8.70 $\pm$ 2.26	18.20 $\pm$ 4.42
Expr	4.27 $\pm$ 1.79	5.40 $\pm$ 0.97	4.60 $\pm$ 1.26	4.80 $\pm$ 1.13
Seq	3.93 $\pm$ 0.80	3.70 $\pm$ 0.67	4.00 $\pm$ 1.25	4.50 $\pm$ 0.97
	Rp-Labels	Rp-Predicted	Rp-True	Rp-NoLabels
Celcycle	15.60 $\pm$ 1.88	17.70 $\pm$ 1.34	14.70 $\pm$ 0.82	16.80 $\pm$ 2.15
Church	27.73 $\pm$ 13.93	24.20 $\pm$ 2.74	17.40 $\pm$ 1.35	25.40 $\pm$ 4.57
Derisi	23.53 $\pm$ 3.76	17.20 $\pm$ 1.69	14.20 $\pm$ 1.32	14.80 $\pm$ 1.75
Eisen	15.73 $\pm$ 1.62	18.40 $\pm$ 1.90	14.80 $\pm$ 0.79	16.70 $\pm$ 1.42
Gasch1	17.67 $\pm$ 1.88	20.20 $\pm$ 2.39	16.30 $\pm$ 0.82	19.90 $\pm$ 2.13
Gasch2	25.47 $\pm$ 7.97	23.30 $\pm$ 2.50	16.30 $\pm$ 1.42	25.50 $\pm$ 2.01
Pheno	26.93 $\pm$ 10.28	31.30 $\pm$ 6.96	20.40 $\pm$ 2.95	22.00 $\pm$ 3.20
Spo	16.40 $\pm$ 2.38	15.90 $\pm$ 1.20	12.90 $\pm$ 0.99	14.80 $\pm$ 1.81
Expr	17.87 $\pm$ 1.55	17.80 $\pm$ 1.48	17.60 $\pm$ 1.17	17.00 $\pm$ 1.33
Seq	16.53 $\pm$ 1.12	16.10 $\pm$ 1.10	15.80 $\pm$ 1.03	15.80 $\pm$ 0.92

Tabela 6.11: Número médio de ciclos necessários até a convergências dos algoritmos *Back-propagation* e *Resilient Back-propagation* nos conjuntos de dados estruturados como grafos

Dataset	Bp-Labels	Bp-Predicted	Bp-True	Bp-NoLabels
Celcycle	11.10 $\pm$ 2.19	18.62 $\pm$ 5.12	5.60 $\pm$ 1.50	22.00 $\pm$ 3.50
Church	11.50 $\pm$ 5.38	20.14 $\pm$ 5.76	5.10 $\pm$ 1.45	29.10 $\pm$ 6.64
Derisi	12.80 $\pm$ 4.49	21.43 $\pm$ 3.87	4.00 $\pm$ 0.94	27.70 $\pm$ 3.47
Eisen	10.50 $\pm$ 6.87	21.40 $\pm$ 3.72	8.20 $\pm$ 1.32	27.60 $\pm$ 6.08
Gasch1	12.00 $\pm$ 5.33	9.10 $\pm$ 3.78	4.80 $\pm$ 0.92	11.90 $\pm$ 2.38
Gasch2	18.80 $\pm$ 9.85	21.60 $\pm$ 5.32	6.20 $\pm$ 1.03	32.20 $\pm$ 4.18
Pheno	7.80 $\pm$ 3.01	26.00 $\pm$ 9.32	10.10 $\pm$ 4.68	18.00 $\pm$ 6.99
Spo	8.00 $\pm$ 4.83	15.20 $\pm$ 3.63	4.20 $\pm$ 1.32	18.70 $\pm$ 2.26
Expr	3.10 $\pm$ 1.59	3.40 $\pm$ 1.26	2.50 $\pm$ 0.53	4.10 $\pm$ 0.99
Seq	4.30 $\pm$ 2.16	3.00 $\pm$ 0.82	2.60 $\pm$ 0.70	3.70 $\pm$ 0.67
	Rp-Labels	Rp-Predicted	Rp-True	Rp-NoLabels
Celcycle	16.50 $\pm$ 2.12	17.70 $\pm$ 1.64	13.40 $\pm$ 0.84	15.50 $\pm$ 1.35
Church	20.50 $\pm$ 2.84	18.80 $\pm$ 1.87	13.60 $\pm$ 0.70	19.10 $\pm$ 1.37
Derisi	26.00 $\pm$ 13.40	19.10 $\pm$ 1.73	13.20 $\pm$ 1.03	14.40 $\pm$ 1.43
Eisen	16.10 $\pm$ 1.20	18.50 $\pm$ 1.72	13.80 $\pm$ 1.03	15.50 $\pm$ 0.71
Gasch1	18.30 $\pm$ 1.83	23.90 $\pm$ 3.03	14.90 $\pm$ 0.99	19.40 $\pm$ 1.90
Gasch2	20.60 $\pm$ 2.01	21.70 $\pm$ 1.83	13.90 $\pm$ 1.20	19.60 $\pm$ 1.07
Pheno	20.30 $\pm$ 1.49	24.40 $\pm$ 3.34	20.70 $\pm$ 1.77	19.30 $\pm$ 1.42
Spo	17.70 $\pm$ 1.49	18.20 $\pm$ 1.23	14.20 $\pm$ 1.32	16.50 $\pm$ 0.97
Expr	21.70 $\pm$ 4.27	22.50 $\pm$ 1.35	19.00 $\pm$ 1.15	19.10 $\pm$ 0.99
Seq	26.80 $\pm$ 3.76	21.00 $\pm$ 0.94	18.00 $\pm$ 0.94	17.90 $\pm$ 0.99

mente nos conjuntos de dados Expr e Seq. Nesses conjuntos de dados, a utilização do algoritmo *Resilient Back-propagation* levou a um número de ciclos de treinamento muito maior do que a utilização do algoritmo *Back-propagation*. Esse maior tempo de treinamento levou a melhores resultados na maioria dos casos, utilizando ambas as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$.

Deve ser observada ainda a grande melhora de desempenho obtida quando os atributos dos exemplos são utilizados também a partir do segundo nível hierárquico. Como esperado, os desempenhos das variações Bp-Labels e Rp-Labels, tanto em hierarquias estruturadas como árvores quanto em hierarquias estruturadas como grafos, foi muito inferior aos resultados obtidos pelas outras variações. Isso torna evidente que, apesar de mostrarem-se muito úteis, as predições sozinhas não representam informação suficiente para um bom desempenho preditivo do segundo nível hierárquico em diante.

A Tabela 6.12 mostra os rankings médios das variações do método HMC-LMLP de acordo com o teste de Friedman. Foram feitos dois testes, considerando separadamente os resultados obtidos com as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Após a aplicação do teste, os valores de p -value obtidos considerando as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$ foram, respectivamente, 8.03×10^{-19} e 6.38×10^{-38} , o que indica claramente que houve diferenças estatisticamente significantes entre os métodos. Para identificar as comparações par a par nas quais há diferenças estatisticamente significantes, o teste post-hoc de Nemenyi foi aplicado. Os diagramas da Figura 6.3 mostram os resultados do teste de Nemenyi considerando todos os pares de variações do método HMC-LMLP. São conectadas as variações em cujas comparações não foram detectadas diferenças estatisticamente significantes.

Tabela 6.12: Rankings médios das variações do método HMC-LMLP de acordo com o teste de Friedman.

Método	$AU(\overline{PRC})$ Ranking	Método	$\overline{AUPRC}_w$ Ranking
Bp-Predicted	2.50	Bp-Predicted	1.85
Rp-Predicted	2.82	Rp-Predicted	2.20
Rp-NoLabels	2.95	Rp-NoLabels	3.72
Bp-NoLabels	3.60	Rp-True	4.02
Rp-True	5.05	Bp-NoLabels	4.02
Bp-True	5.45	Bp-True	5.17
Rp-Labels	6.57	Rp-Labels	7.42
Bp-Labels	7.05	Bp-Labels	7.57

Como o método HMC-LMLP utiliza a estratégia de classificação *Local Classifier per Level* (LCL), que associa um classificador a cada nível hierárquico, todas as suas variações foram comparadas quanto aos desempenhos obtidos em cada nível. As Tabelas do Apêndice A (A.1, A.2, A.3, A.4, A.5 e A.6) apresentam as médias das áreas abaixo das curvas PR individuais ($AUPRC$) obtidas em cada classe, por nível hierárquico.

Experimentos comparando as variações Bp-Predicted, Rp-Predicted e métodos da literatura

Após comparar as diferentes variações do método HMC-LMLP entre si, as variações utilizando as predições para complementar os vetores de atributos dos exemplos (Bp-Predicted e Rp-Predicted) foram comparadas com diferentes métodos da literatura.

Figura 6.3: Comparação para a par das variações do método HMC-LMLP utilizando o teste estatístico de Nemenyi. (a) resultados com a medida $AU(\overline{PRC})$; (b) resultados com a medida $\overline{AUPRC}_w$.

Comparando os desempenhos dos métodos nos conjuntos de dados estruturados como árvores (Tabela 6.13), observou-se que o método HMC-LMLP obteve os melhores desempenhos na maioria dos conjuntos de dados, considerando tanto a avaliação com a medida $AU(\overline{PRC})$ quanto com a medida $\overline{AUPRC}_w$.

Analisando os resultados obtidos nos conjuntos de dados estruturados como grafos (Tabela 6.14) observa-se um maior equilíbrio entre os resultados obtidos pelos métodos considerando a medida $AU(\overline{PRC})$, sendo os melhores resultados distribuídos entre os métodos HMC-LMLP, Clus-HMC e Clus-HSC. Como mencionado anteriormente, quando todas as variações do método HMC-LMLP foram comparadas, os resultados dos métodos Bp-Predicted e Rp-Predicted poderiam ser melhores se todos os relacionamentos hierárquicos entre as classes tivessem sido considerados (lembrando que alguns relacionamentos não foram considerados devido à adaptação feita nas hierarquias para a aplicação do método HMC-LMLP).

Ainda nos conjuntos de dados estruturados como grafos, considerando a avaliação com a medida $\overline{AUPRC}_w$, observa-se que os melhores resultados foram obtidos pelo método Clus-HMC. Como a medida $\overline{AUPRC}_w$ considera as frequências das classes nos conjuntos de dados, os resultados sugerem que os desempenhos das variações Bp-Predicted e Rp-Predicted foram melhores na maioria das classes menos frequentes dos conjuntos de dados. Isso acontece porque a avaliação ponderando os desempenhos individuais nas classes diminui muito os valores de $AUPRC$ individuais obtidos em classes pouco frequentes.

A Tabela 6.15 mostra os rankings médios das variações Bp-Predicted, Rp-Predicted e métodos da literatura de acordo com o teste de Friedman. Assim como nas comparações envolvendo todas as variações do método HMC-LMLP, foram feitos testes considerando separadamente os resultados obtidos com as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Após a aplicação do teste, os valores de p -value obtidos considerando as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$ foram, respectivamente, 2.51×10^{-15} e 6.15×10^{-10} , o que indica claramente que houve diferenças estatisticamente

Tabela 6.13: Comparação dos métodos Bp-Predicted e Rp-Predicted e métodos da literatura em conjuntos de dados estruturados como árvores

Dados	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
$AU(\overline{PRC})$						
Cellcycle	0.207 ± 0.001	0.202 ± 0.002	0.172	0.111	0.106	0.155 ± 0.008
Church	0.173 ± 0.001	0.170 ± 0.001	0.170	0.131	0.128	0.165 ± 0.003
Derisi	0.183 ± 0.001	0.177 ± 0.001	0.175	0.094	0.089	0.149 ± 0.008
Eisen	0.245 ± 0.002	0.239 ± 0.002	0.204	0.127	0.132	0.181 ± 0.007
Gasch1	0.236 ± 0.002	0.235 ± 0.002	0.205	0.106	0.104	0.173 ± 0.008
Gasch2	0.211 ± 0.001	0.206 ± 0.002	0.195	0.121	0.119	0.152 ± 0.007
Pheno	0.159 ± 0.002	0.155 ± 0.002	0.160	0.152	0.149	0.161 ± 0.004
Spo	0.186 ± 0.002	0.178 ± 0.001	0.186	0.103	0.098	0.177 ± 0.004
Expr	0.243 ± 0.002	0.246 ± 0.004	0.210	0.127	0.123	0.180 ± 0.007
Seq	0.236 ± 0.002	0.249 ± 0.003	0.211	0.091	0.095	0.186 ± 0.008
Média	0.208	0.206	0.189	0.116	0.114	0.168
$\overline{AUPRC}_w$						
Cellcycle	0.184 ± 0.003	0.181 ± 0.002	0.142	0.146	0.146	0.133 ± 0.004
Church	0.131 ± 0.001	0.130 ± 0.001	0.129	0.127	0.128	0.123 ± 0.001
Derisi	0.146 ± 0.002	0.142 ± 0.002	0.137	0.125	0.122	0.132 ± 0.002
Eisen	0.221 ± 0.003	0.220 ± 0.004	0.183	0.169	0.173	0.151 ± 0.007
Gasch1	0.213 ± 0.001	0.218 ± 0.002	0.176	0.154	0.153	0.154 ± 0.004
Gasch2	0.185 ± 0.001	0.180 ± 0.002	0.156	0.148	0.147	0.142 ± 0.003
Pheno	0.125 ± 0.002	0.120 ± 0.002	0.124	0.125	0.127	0.121 ± 0.001
Spo	0.152 ± 0.002	0.145 ± 0.002	0.153	0.139	0.139	0.139 ± 0.003
Expr	0.236 ± 0.002	0.233 ± 0.003	0.179	0.167	0.167	0.159 ± 0.004
Seq	0.220 ± 0.003	0.240 ± 0.002	0.183	0.151	0.154	0.155 ± 0.002
Média	0.181	0.181	0.156	0.145	0.146	0.141

Tabela 6.14: Comparação dos métodos Bp-Predicted e Rp-Predicted e métodos da literatura em conjuntos de dados estruturados como grafos

Dados	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
$AU(\overline{PRC})$						
Cellcycle	0.361 ± 0.001	0.364 ± 0.001	0.357	0.371	0.252	0.325 ± 0.008
Church	0.341 ± 0.001	0.347 ± 0.001	0.348	0.397	0.289	0.334 ± 0.001
Derisi	0.343 ± 0.001	0.344 ± 0.001	0.355	0.349	0.218	0.321 ± 0.007
Eisen	0.403 ± 0.002	0.404 ± 0.001	0.380	0.365	0.270	0.373 ± 0.011
Gasch1	0.380 ± 0.002	0.378 ± 0.003	0.371	0.351	0.239	0.352 ± 0.008
Gasch2	0.369 ± 0.001	0.372 ± 0.001	0.365	0.378	0.267	0.334 ± 0.016
Pheno	0.318 ± 0.002	0.314 ± 0.002	0.337	0.416	0.316	0.336 ± 0.002
Spo	0.342 ± 0.001	0.341 ± 0.001	0.352	0.371	0.213	0.326 ± 0.008
Expr	0.372 ± 0.003	0.377 ± 0.003	0.368	0.351	0.249	0.343 ± 0.007
Seq	0.370 ± 0.002	0.382 ± 0.002	0.386	0.282	0.197	0.371 ± 0.007
Média	0.360	0.362	0.362	0.363	0.251	0.341
$\overline{AUPRC}_w$						
Cellcycle	0.304 ± 0.002	0.304 ± 0.002	0.335	0.275	0.267	0.324 ± 0.003
Church	0.257 ± 0.001	0.257 ± 0.002	0.316	0.248	0.247	0.311 ± 0.002
Derisi	0.267 ± 0.002	0.263 ± 0.002	0.321	0.248	0.246	0.314 ± 0.002
Eisen	0.345 ± 0.002	0.346 ± 0.003	0.362	0.303	0.294	0.349 ± 0.004
Gasch1	0.328 ± 0.004	0.329 ± 0.006	0.353	0.290	0.282	0.337 ± 0.002
Gasch2	0.311 ± 0.001	0.309 ± 0.001	0.347	0.282	0.278	0.334 ± 0.006
Pheno	0.231 ± 0.001	0.232 ± 0.001	0.299	0.239	0.238	0.299 ± 0.002
Spo	0.268 ± 0.002	0.261 ± 0.001	0.324	0.254	0.254	0.318 ± 0.004
Expr	0.332 ± 0.003	0.331 ± 0.002	0.353	0.286	0.284	0.326 ± 0.004
Seq	0.330 ± 0.003	0.347 ± 0.002	0.373	0.283	0.279	0.344 ± 0.002
Média	0.297	0.298	0.338	0.271	0.267	0.326

significantes entre os métodos. Para identificar as comparações par a par nas quais há diferenças estatisticamente significantes, o teste post-hoc de Nemenyi foi aplicado. Os diagramas da Figura 6.4 mostram os resultados do teste de Nemenyi considerando todos os pares de comparações. São conectados os métodos em cujas comparações não foram detectadas diferenças estatisticamente significantes.

Tabela 6.15: Rankings médios das variações Bp-Predicted, Rp-Predicted e métodos da literatura de acordo com o teste de Friedman.

Método	$AU(\overline{PRC})$ Ranking	Método	$\overline{AUPRC}_w$ Ranking
Bp-Predicted	2.22	Clus-HMC	2.07
Rp-Predicted	2.37	Bp-Predicted	2.52
Clus-HMC	2.60	Rp-Predicted	2.90
Clus-HSC	3.90	hmAnt-Miner	3.75
hmAnt-Miner	4.05	Clus-HSC	4.67
Clus-SC	5.85	Clus-SC	5.07

Figura 6.4: Comparação para a par das variações Bp-Predicted, Rp-Predicted e métodos da literatura utilizando o teste estatístico de Nemenyi. (a) resultados com a medida $AU(\overline{PRC})$; (b) resultados com a medida $\overline{AUPRC}_w$.

Assim como foi feito na comparação de todas as variações do método **HMC-LMLP**, os desempenhos das variações Bp-Predicted, Rp-Predicted, e de todos os métodos da literatura, também foram comparados em cada nível. As Tabelas B.1, B.2, B.3, B.4 e B.5 do Apêndice B apresentam esses resultados.

Além de analisar os resultados nível a nível, as variações Bp-Predicted e Rp-Predicted também foram comparadas com os métodos na literatura quanto aos seus desempenhos em classes específicas das hierarquias. Esses resultados são apresentados na Tabelas C.1, C.2, C.3, C.4, C.5, C.6, C.7, C.8, C.9 e C.10 do Apêndice C.

Os resultados apresentando os desempenhos nível a nível (Apêndice B) e em classes específicas (Apêndice C) nas hierarquias estruturadas como grafo podem parecer estranhos, pois parecem contrastar com os resultados apresentados na Tabela 6.14. Apesar de apresentar os melhores resultados nível a nível, e também obter os melhores resultados na maioria das classes investigadas, as variações Bp-Predicted e Rp-Predicted não apresentam os melhores resultados de maneira geral (Tabela 6.14). No entanto, essa diferença de resultados pode ser explicada

se forem observadas as quantidades de classes nas quais os métodos obtiveram os melhores resultados. Essa comparação é feita na Tabela 6.16.

Tabela 6.16: Comparação par a par das variações Bp-Predicted, Rp-Predicted e métodos da literatura quanto ao número de vitórias em classes específicas das hierarquias estruturadas como grafos

Conjunto de dados	Bp-Predicted x Clus-HMC	Bp-Predicted x Clus-HSC	Bp-Predicted x Clus-SC	Bp-Predicted x <i>hmAnt-Miner</i>
Cellcycle	931 x 1907	1228 x 1610	1967 x 871	578 x 2260
Church	495 x 2343	743 x 2095	1064 x 1774	480 x 2358
Derisi	491 x 2341	1173 x 1659	1848 x 984	360 x 2472
Eisen	768 x 1671	883 x 1556	1141 x 1298	531 x 1908
Gasch1	796 x 2042	1295 x 1543	1888 x 950	527 x 2311
Gasch2	794 x 2051	1186 x 1659	1890 x 955	406 x 2439
Pheno	166 x 1947	193 x 1920	338 x 1775	313 x 1800
Spo	509 x 2323	928 x 1904	1305 x 1527	452 x 2380
Expr	941 x 1904	1388 x 1457	1946 x 899	627 x 2218
Seq	657 x 2192	1301 x 1548	1778 x 1071	461 x 2388
Conjunto de dados	Rp-Predicted x Clus-HMC	Rp-Predicted x Clus-HSC	Rp-Predicted x Clus-SC	Rp-Predicted x <i>hmAnt-Miner</i>
Cellcycle	1425 x 1413	1635 x 1203	2008 x 830	1193 x 1645
Church	886 x 1952	1047 x 1791	1185 x 1653	743 x 2095
Derisi	863 x 1969	1359 x 1473	1825 x 1007	628 x 2204
Eisen	1240 x 1199	1345 x 1094	1420 x 1019	1129 x 1310
Gasch1	1321 x 1517	1642 x 1196	2042 x 796	1077 x 1761
Gasch2	1466 x 1379	1633 x 1212	1955 x 890	1080 x 1765
Pheno	360 x 1753	451 x 1662	490 x 1623	495 x 1618
Spo	792 x 2040	1120 x 1712	1310 x 1522	665 x 2167
Expr	1164 x 1681	1553 x 1292	2100 x 745	950 x 1895
Seq	1327 x 1522	1674 x 1175	2095 x 754	1183 x 1666

Como pode ser observado na Tabela 6.16, os métodos Clus-HMC, Clus-HSC e *hmAnt-Miner* obtiveram melhores resultados em mais classes do que as variações Bp-Predicted e Rp-Predicted. Essa diferença é mais acentuada nas comparações envolvendo o algoritmo *Back-propagation*, e ajuda a explicar os resultados da Tabela 6.14.

Ainda observando os resultados das Tabela 6.16 e 6.14, há casos em que mesmo com a diferença entre o número de vitórias de dois métodos sendo muito grande, o método que obteve valores maiores de *AUPRC* para menos classes obteve um maior valor de $AU(\overline{PRC})$. Um exemplo desse caso está na comparação envolvendo a variação Bp-Predicted e o método Clus-HMC no conjunto de dados Cellcycle. Nesse conjunto de dados, a variação Bp-Predicted obteve os melhores valores de *AUPRC* em 931 classes, enquanto o método Clus-HMC foi melhor em 1907 classes. Ainda assim, a variação Bp-Predicted obteve um valor de $AU(\overline{PRC})$ igual a 0.361, enquanto o método Clus-HMC obteve um valor de 0.357 (primeira linha da tabela 6.14). Essa diferença pode ser explicada se for considerado o número de classes preditas por cada método conforme os valores de limiares são aplicados para a obtenção dos pontos que formarão as curvas precisão-revoação. Valores altos de limiares fazem com que mais classes sejam incluídas nos conjuntos de classes preditas pelos métodos da literatura do que no conjunto de classes preditas pela variação Bp-Predicted. Assim, para um mesmo valor de limiar, a tendência é que os métodos da literatura obtenham valores de revocação e precisão menores do que a variação Bp-Predicted. Ainda, observou-se que para um mesmo valor de revocação, as variações do método HMC-LMLP obtiveram maiores valores de precisão. Isso é indicativo de um maior número de classes preditas pelos métodos da literatura. Isso pode ser verificado nos gráficos comparando as curvas precisão-revoação de todos os métodos (ver Seção 6.6).

Com a finalidade de exemplificar o efeito da aplicação de vários valores de limiares, na Tabela 6.17 é apresentada uma comparação dos valores de precisão e revocação dos métodos Bp-Predicted, Rp-Predicted e Clus-HMC (considerado o método estado-da-arte na literatura) conforme os valores de limiares são aumentados. Considerou-se o conjunto de dados Cellcycle cujas classes são estruturadas como árvores. Os valores mostrados para as variações Bp-Predicted e Rp-Predicted correspondem aos valores obtidos nas execuções que levaram aos melhores valores de $AU(\overline{PRC})$ no conjunto de validação.

Tabela 6.17: Valores de Precisão e Revocação obtidos no conjunto de dados Cellcycle estruturado como árvore

Limiares	Bp-Predicted		Rp-Predicted		Clus-HMC	
	Precisão	Revocação	Precisão	Revocação	Precisão	Revocação
0.0	0.018	1.0	0.018	1.0	0.018	1.0
4.0	0.105	0.684	0.077	0.789	0.100	0.685
8.0	0.172	0.470	0.132	0.592	0.161	0.475
12.0	0.226	0.361	0.190	0.434	0.202	0.355
16.0	0.268	0.284	0.243	0.318	0.236	0.275
20.0	0.304	0.231	0.293	0.227	0.269	0.206
24.0	0.336	0.188	0.345	0.161	0.302	0.143
28.0	0.358	0.151	0.392	0.114	0.329	0.094
32.0	0.388	0.126	0.435	0.081	0.350	0.072
36.0	0.429	0.106	0.483	0.056	0.365	0.054
40.0	0.456	0.089	0.543	0.041	0.394	0.030
44.0	0.497	0.075	0.619	0.030	0.450	0.024
48.0	0.531	0.061	0.672	0.019	0.460	0.018
52.0	0.572	0.051	0.720	0.012	0.479	0.013
56.0	0.604	0.041	0.806	0.007	0.528	0.012
60.0	0.638	0.033	0.828	0.004	0.568	0.011
64.0	0.661	0.025	0.867	0.003	0.591	0.010
68.0	0.697	0.021	0.919	0.003	0.576	0.007
72.0	0.724	0.015	0.935	0.002	0.574	0.006
76.0	0.791	0.011	0.960	0.002	0.610	0.005
80.0	0.842	0.008	1.0	0.001	0.769	0.003
84.0	0.866	0.005	0.0	0.0	0.784	0.003
88.0	0.872	0.003	0.0	0.0	0.784	0.003
92.0	0.882	0.001	0.0	0.0	0.808	0.003
96.0	1.000	0.00009	0.0	0.0	0.968	0.003
100.0	0.0	0.0	0.0	0.0	0.0	0.0

Como pode ser observado na Tabela 6.17, as variações Bp-Predicted e Rp-Predicted obtiveram, na maioria dos limiares utilizados, maiores valores de precisão e revocação do que o método Clus-HMC. Ainda, para um mesmo valor de revocação, as variações do método HMC-LMLP obtiveram valores de precisão mais altos do que o método Clus-HMC. Exemplificando, para o valor de revocação 0.030, o método Clus-HMC obteve o valor 0.394 de precisão, enquanto a variação Rp-Predicted obteve o valor 0.619. Com um valor aproximado de revocação (0.033), a variação Bp-Predicted obteve o valor 0.638 de precisão.

6.5 Experimentos com o Método HMC-GA

Nesta seção são apresentados os experimentos realizados com o método HMC-GA. Como o método é baseado na abordagem global, diferentemente da análise experimental realizada com os resultados do método HMC-LMLP, optou-se por analisar os resultados apenas de uma maneira global. Essa escolha foi baseada no fato da característica da abordagem global ser

justamente a realização de predições considerando todos os níveis simultaneamente. Adicionalmente, não foi feita nenhuma adaptação nos conjuntos de dados organizados como grafos para a aplicação do método **HMC-GA**, já que não há necessidade de definição clara dos níveis das classes para a aplicação do método. Sendo assim, não faz sentido uma análise nível a nível nesse caso.

6.5.1 Parâmetros

A Tabela 6.18 apresenta os valores dos parâmetros utilizados no método **HMC-GA**. O método foi executado com três configurações diferentes para o parâmetro *Tamanho da população* (p). O algoritmo genético foi executado com 100, 500 e 1000 indivíduos na população.

Tabela 6.18: Parâmetros utilizados no método HMC-GA

Parâmetro	Valor
Tamanho da população (p)	100/500/1000
Taxa de elitismo (e)	1%
Taxa de mutação (mr)	40%
Taxa de <i>crossover</i> (cr)	90%
Probabilidade de utilizar um atributo na inicialização (pt)	$ D /pt = 5$
Número de gerações (G)	100
Tamanho do torneio (t)	2
Número máximo de exemplos não cobertos ($maxNaoCob$)	10
Número mínimo de exemplos cobertos por regra ($minCob$)	10
Número máximo de exemplos cobertos por regra ($maxCob$)	300

O parâmetro pt refere-se à probabilidade de um atributo ser utilizado quando criando os antecedentes da população inicial (procedimento de semeadura - Capítulo 5, Seção 5.2). Uma probabilidade baixa gera antecedentes menores que têm mais chances de cobrir exemplos. Essa mesma probabilidade também é utilizada quando é feita a mutação dos **FLAGS** dos antecedentes. Durante essa mutação, a utilização de um valor alto de probabilidade ativaria muitos testes no antecedente, resultando em uma regra que cobriria poucos ou nenhum exemplo.

Ainda sobre o parâmetro pt , vê-se na Tabela 6.18 que ele é dado pela expressão $|A|/pt = 5$, na qual A é o conjunto de atributos de um conjunto de dados. Assim, é escolhido um valor de probabilidade pt que resulte na utilização de, em média, cinco atributos em cada antecedente das regras geradas.

O *Número máximo de exemplos não cobertos* ($maxNaoCob$) refere-se ao número máximo de exemplos que podem ser deixados sem serem cobertos por nenhuma regra. Se ao final do processo evolutivo, restar no conjunto de treinamento um número de exemplos menor ou igual ao número especificado, o procedimento de cobertura sequencial é finalizado. Os exemplos não cobertos são então classificados na regra padrão.

O parâmetro $minCob$ é utilizado como limite inferior para o número de exemplos cobertos pelas regras geradas. Assim, evita-se que sejam criadas regras muito específicas. Esse parâmetro, junto com o parâmetro $maxCob$, também é utilizado no procedimento de busca local do método **HMC-GA** (Capítulo 5, Algoritmo 9). O objetivo é fazer com que as regras geradas não sejam nem muito específicas e nem muito gerais.

O parâmetro G refere-se ao número máximo de gerações evoluídas pelo algoritmo genético para se obter uma regra de classificação. Após esse número máximo de gerações, a melhor regra do conjunto de regras evoluídas é salva e os exemplos cobertos por ela são retirados do conjunto de treinamento. O procedimento de evolução é então reiniciado com novas regras iniciais.

Os parâmetros apresentados na Tabela 6.18 foram baseados no trabalho de [Carvalho et al. \(2011\)](#), que desenvolveu um método baseado na abordagem *Local Classifier per Node (LCN)* utilizando algoritmos genéticos. Optou-se pela utilização desse trabalho como base por tratar-se do único trabalho, até o momento, que utilizava algoritmos genéticos para esse tipo de classificação. Ainda, foram realizados experimentos preliminares com o método [HMC-GA](#) utilizando diferentes taxas de mutação e tamanhos de torneio. Percebeu-se que menores taxas de mutação e mais indivíduos no torneio diminuíram a diversidade das regras geradas, fazendo com que a evolução terminasse prematuramente, diminuindo o desempenho de classificação.

6.5.2 Experimentos

Os resultados das medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$, para hierarquias estruturadas como árvores e grafos, são apresentados nas Tabelas 6.19 e 6.20. O método [HMC-GA](#) foi executado com três variações no número de indivíduos (regras) da população: 100, 500 e 1000 indivíduos. Como pode ser observado, nos conjuntos de dados com os maiores números de atributos (Gasch1, Expr e Seq), houve uma tendência de melhora no desempenho de classificação conforme o número de indivíduos foi aumentado. Essa melhora é mais acentuada nos conjuntos de dados Expr e Seq.

Considerando os experimentos nos conjuntos de dados estruturados como árvores, os melhores desempenhos na medida $AU(\overline{PRC})$ foram obtidos pelos métodos Clus-HMC e [HMC-GA](#). Se for considerada a medida $\overline{AUPRC}_w$, o método [HMC-GA](#) é superado por todos os métodos da literatura, apesar de apresentar um desempenho competitivo se comparado com o método *hmAnt-Miner*. Isso é indicativo que esses métodos foram melhores que o Algoritmo Genético nas classes mais frequentes dos conjuntos de dados, e que o método [HMC-GA](#) foi melhor ou mais competitivo nas classes menos frequentes.

Nos conjuntos de dados estruturados como grafos, os métodos Clus-HMC e Clus-HSC obtiveram os melhores resultados na medida $AU(\overline{PRC})$. Logo em seguida, o melhor desempenho foi obtido pelo método [HMC-GA](#), superando os métodos *hmAnt-Miner* e Clus-SC. No entanto, novamente o método [HMC-GA](#) obteve os piores desempenhos na medida $\overline{AUPRC}_w$, indicativo de que o método proposto obteve bons desempenhos, ou desempenhos mais competitivos com os métodos da literatura, em classes pouco frequentes nos conjuntos de dados.

A Tabela 6.21 mostra os rankings médios do método [HMC-GA](#) quando utilizado com tamanhos de população igual a 100, 500 e 1000 indivíduos, e também dos métodos da literatura, de acordo com o teste de Friedman. Foram feitos testes considerando separadamente os resultados obtidos com as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Após a aplicação do teste, os valores de p -value obtidos considerando as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$ foram, respectivamente, 4.39×10^{-20} e 4.44×10^{-26} , o que indica claramente que houve diferenças estatisticamente significantes entre

Tabela 6.19: Comparação dos métodos HMC-GA e métodos da literatura em conjuntos de dados estruturados como árvores

Dados	100 Indivíduos	HMC-GA 500 Indivíduos	1000 Indivíduos	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
$AU(\overline{PRC})$							
Cellcycle	0.164 ± 0.004	0.163 ± 0.007	0.167 ± 0.008	0.172	0.111	0.106	0.155 ± 0.008
Church	0.156 ± 0.008	0.157 ± 0.011	0.155 ± 0.010	0.170	0.131	0.128	0.165 ± 0.003
Derisi	0.168 ± 0.005	0.167 ± 0.004	0.168 ± 0.004	0.175	0.094	0.089	0.149 ± 0.008
Eisen	0.185 ± 0.008	0.190 ± 0.005	0.200 ± 0.007	0.204	0.127	0.132	0.181 ± 0.007
Gasch1	0.188 ± 0.006	0.191 ± 0.007	0.192 ± 0.007	0.205	0.106	0.104	0.173 ± 0.008
Gasch2	0.173 ± 0.009	0.174 ± 0.009	0.170 ± 0.010	0.195	0.121	0.119	0.152 ± 0.007
Pheno	0.160 ± 0.003	0.156 ± 0.003	0.159 ± 0.001	0.160	0.152	0.149	0.161 ± 0.004
Spo	0.173 ± 0.006	0.173 ± 0.008	0.173 ± 0.008	0.186	0.103	0.098	0.177 ± 0.004
Expr	0.179 ± 0.008	0.186 ± 0.010	0.193 ± 0.006	0.210	0.127	0.123	0.180 ± 0.007
Seq	0.172 ± 0.007	0.178 ± 0.016	0.184 ± 0.015	0.211	0.091	0.095	0.186 ± 0.008
Média	0.172	0.173	0.176	0.189	0.116	0.114	0.168
$\overline{AUPRC}_w$							
Cellcycle	0.132 ± 0.004	0.136 ± 0.006	0.140 ± 0.006	0.142	0.146	0.146	0.133 ± 0.004
Church	0.122 ± 0.002	0.122 ± 0.003	0.123 ± 0.003	0.129	0.127	0.128	0.123 ± 0.001
Derisi	0.125 ± 0.005	0.124 ± 0.005	0.125 ± 0.004	0.137	0.125	0.122	0.132 ± 0.002
Eisen	0.144 ± 0.009	0.155 ± 0.009	0.158 ± 0.008	0.183	0.169	0.173	0.151 ± 0.007
Gasch1	0.148 ± 0.006	0.153 ± 0.005	0.153 ± 0.004	0.176	0.154	0.153	0.154 ± 0.004
Gasch2	0.136 ± 0.007	0.139 ± 0.003	0.137 ± 0.005	0.156	0.148	0.147	0.142 ± 0.003
Pheno	0.118 ± 0.003	0.118 ± 0.002	0.118 ± 0.002	0.124	0.125	0.127	0.121 ± 0.001
Spo	0.130 ± 0.012	0.136 ± 0.006	0.137 ± 0.004	0.153	0.139	0.139	0.139 ± 0.003
Expr	0.140 ± 0.009	0.149 ± 0.013	0.155 ± 0.004	0.179	0.167	0.167	0.159 ± 0.004
Seq	0.131 ± 0.008	0.137 ± 0.019	0.143 ± 0.018	0.183	0.151	0.154	0.155 ± 0.002
Média	0.133	0.137	0.139	0.156	0.145	0.146	0.141

Tabela 6.20: Comparação dos métodos HMC-GA e métodos da literatura em conjuntos de dados estruturados como grafos

Dados	100 Indivíduos	HMC-GA 500 Indivíduos	1000 Indivíduos	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
$AU(\overline{PRC})$							
Cellcycle	0.341 ± 0.009	0.344 ± 0.007	0.339 ± 0.008	0.357	0.371	0.252	0.325 ± 0.008
Church	0.341 ± 0.005	0.340 ± 0.005	0.338 ± 0.008	0.348	0.397	0.289	0.334 ± 0.001
Derisi	0.344 ± 0.003	0.342 ± 0.005	0.343 ± 0.005	0.355	0.349	0.218	0.321 ± 0.007
Eisen	0.380 ± 0.004	0.375 ± 0.009	0.377 ± 0.011	0.380	0.365	0.270	0.373 ± 0.011
Gasch1	0.361 ± 0.003	0.361 ± 0.009	0.362 ± 0.006	0.371	0.351	0.239	0.352 ± 0.008
Gasch2	0.351 ± 0.007	0.354 ± 0.008	0.352 ± 0.004	0.365	0.378	0.267	0.334 ± 0.016
Pheno	0.331 ± 0.002	0.331 ± 0.004	0.332 ± 0.004	0.337	0.416	0.316	0.336 ± 0.002
Spo	0.345 ± 0.008	0.349 ± 0.006	0.350 ± 0.005	0.352	0.371	0.213	0.326 ± 0.008
Expr	0.361 ± 0.007	0.361 ± 0.004	0.364 ± 0.007	0.368	0.351	0.249	0.343 ± 0.007
Seq	0.359 ± 0.007	0.358 ± 0.016	0.366 ± 0.011	0.386	0.282	0.197	0.371 ± 0.007
Média	0.351	0.351	0.352	0.362	0.363	0.251	0.341
$\overline{AUPRC}_w$							
Cellcycle	0.256 ± 0.008	0.264 ± 0.004	0.263 ± 0.004	0.335	0.275	0.267	0.324 ± 0.003
Church	0.248 ± 0.003	0.251 ± 0.001	0.249 ± 0.003	0.316	0.248	0.247	0.311 ± 0.002
Derisi	0.246 ± 0.004	0.246 ± 0.003	0.247 ± 0.003	0.321	0.248	0.246	0.314 ± 0.002
Eisen	0.282 ± 0.012	0.294 ± 0.007	0.292 ± 0.007	0.362	0.303	0.294	0.349 ± 0.004
Gasch1	0.272 ± 0.009	0.281 ± 0.006	0.284 ± 0.005	0.353	0.290	0.282	0.337 ± 0.002
Gasch2	0.263 ± 0.005	0.271 ± 0.005	0.272 ± 0.004	0.347	0.282	0.278	0.334 ± 0.006
Pheno	0.233 ± 0.004	0.235 ± 0.003	0.234 ± 0.005	0.299	0.239	0.238	0.299 ± 0.002
Spo	0.251 ± 0.008	0.257 ± 0.004	0.260 ± 0.004	0.324	0.254	0.254	0.318 ± 0.004
Expr	0.279 ± 0.014	0.283 ± 0.005	0.289 ± 0.004	0.353	0.286	0.284	0.326 ± 0.004
Seq	0.264 ± 0.016	0.264 ± 0.023	0.279 ± 0.019	0.373	0.283	0.279	0.344 ± 0.002
Média	0.259	0.265	0.267	0.338	0.271	0.267	0.326

os métodos. Para identificar as comparações par a par nas quais há diferenças estatisticamente significantes, o teste post-hoc de Nemenyi foi aplicado. Os diagramas da Figura 6.5 mostram os resultados do teste de Nemenyi considerando todos os pares de comparações. São conectados os métodos em cujas comparações não foram detectadas diferenças estatisticamente significantes.

Tabela 6.21: Rankings médios do método HMC-GA e métodos da literatura de acordo com o teste de Friedman.

Método	$AU(\overline{PRC})$ Ranking	Método	$\overline{AUPRC}_w$ Ranking
Clus-HMC	1.35	Clus-HMC	1.22
HMC-GA (1000 indivíduos)	3.27	hmAnt-Miner	2.87
HMC-GA (100 indivíduos)	3.75	Clus-HSC	3.17
HMC-GA (500 indivíduos)	3.77	Clus-SC	3.97
hmAnt-Miner	4.35	HMC-GA (1000 indivíduos)	4.70
Clus-HSC	4.60	HMC-GA (500 indivíduos)	5.42
Clus-SC	6.90	HMC-GA (100 indivíduos)	6.62

Figura 6.5: Comparação para a par do método HMC-GA e métodos da literatura utilizando o teste estatístico de Nemenyi. (a) resultados com a medida $AU(\overline{PRC})$; (b) resultados com a medida $\overline{AUPRC}_w$.

Como pode ser observado na Figura 6.5(a), o método Clus-HMC foi estatisticamente superior a todos os outros métodos, com exceção do método HMC-GA utilizando 1000 indivíduos. Também não foram detectadas diferenças estatisticamente significantes entre os métodos HMC-GA, Clus-HSC e hmAnt-Miner. Ainda, todos os métodos foram estatisticamente superiores ao método Clus-SC. Apesar do teste estatístico não ter encontrado diferença estatisticamente significativa entre os métodos HMC-GA com 1000 indivíduos e Clus-HMC, é necessário reconhecer que se um teste não paramétrico fosse aplicado considerando apenas esses dois métodos, a diferença seria estatisticamente significativa, já que o método Clus-HMC ganhou do método HMC-GA com 1000 indivíduos em todos os conjuntos de dados (Tabelas 6.19 e 6.20).

Considerando a medida $\overline{AUPRC}_w$ (Figura 6.5), tem-se que o método Clus-HMC superou estatisticamente o método HMC-GA. Por outro lado, em alguns casos não foram detectadas diferenças estatisticamente significantes na comparação dos resultados obtidos pelo método HMC-GA e pelos outros métodos.

Apesar de o desempenho preditivo do método **HMC-GA** não ter superado o desempenho do método estado-da-arte na literatura, os resultados podem ser considerados animadores se forem consideradas as quantidades de regras e os tamanhos das regras geradas pelos métodos. O número de regras geradas por cada método é apresentado na Tabela 6.22, e o tamanho das regras é apresentado na Tabela 6.23. Nessas tabelas, o número de regras geradas pelos métodos Clus-HMC, Clus-HSC e Clus-SC é dado pelo número de folhas das árvores de decisão, já que cada caminho entre o nó raiz e uma folha pode ser considerado uma regra. Para os métodos Clus-HSC e Clus-SC, como esses são métodos locais, são apresentados o número total de regras geradas por cada árvore de decisão, e também, entre parênteses, o número médio de regras de cada árvore individual. Como os métodos **HMC-GA** e *hmAnt-Miner* não são determinísticos, o número de regras apresentado é a média do número de regras obtidas em cada execução.

Tabela 6.22: Número de regras geradas pelo método HMC-GA e métodos da literatura

Dados	100 Indivíduos	HMC-GA 500 Indivíduos	1000 Indivíduos	Clus-HMC	Clus-HSC	Clus-SC	<i>hmAnt-Miner</i>
Conjuntos de dados estruturados como árvore							
Cellcycle	17.30 ± 6.360	23.30 ± 8.220	22.50 ± 11.41	24	4037 (8.1)	9671 (19.4)	25.10 ± 5.626
Church	18.30 ± 7.917	17.40 ± 12.39	19.80 ± 9.283	17	2221 (4.5)	4186 (8.4)	8.10 ± 1.370
Derisi	9.50 ± 2.718	11.10 ± 3.510	11.00 ± 4.163	4	3520 (7.1)	7807 (15.6)	26.60 ± 6.620
Eisen	12.40 ± 4.766	17.40 ± 7.090	12.10 ± 2.998	29	2995 (6.5)	6311 (13.7)	17.50 ± 4.790
Gaschl	14.30 ± 6.799	17.30 ± 4.644	16.30 ± 7.616	10	4761 (9.5)	10447 (20.9)	28.10 ± 3.928
Gasch2	17.50 ± 9.289	20.50 ± 9.132	25.50 ± 9.755	26	3756 (7.5)	7850 (15.7)	36.90 ± 14.487
Pheno	3.90 ± 0.994	4.30 ± 0.823	3.80 ± 1.033	8	777 (1.7)	1238 (2.7)	4.90 ± 0.738
Spo	9.30 ± 4.347	12.20 ± 5.827	14.80 ± 7.376	6	3623 (7.3)	8527 (17.1)	12.80 ± 3.882
Expr	13.50 ± 6.671	17.20 ± 10.99	19.30 ± 7.846	12	4711 (9.4)	10262 (20.6)	28.60 ± 4.452
Seq	11.30 ± 6.701	12.50 ± 5.759	13.30 ± 8.499	14	4923 (9.9)	10443 (20.9)	15.30 ± 3.945
Média	12.73	15.32	15.84	15	3532.4 (7.15)	7674.2 (15.5)	20.39
Conjuntos de dados estruturados como grafo							
Cellcycle	18.60 ± 7.367	18.80 ± 7.177	27.70 ± 7.258	21	19085 (3.3)	36260 (8.8)	39.30 ± 8.138
Church	14.40 ± 6.059	18.70 ± 7.212	15.80 ± 9.235	7	12368 (2.1)	16049 (3.9)	25.90 ± 11.210
Derisi	9.40 ± 2.319	11.30 ± 3.802	11.50 ± 4.836	10	16693 (2.9)	31175 (7.6)	32.90 ± 4.095
Eisen	10.80 ± 4.962	20.90 ± 6.724	18.00 ± 9.080	37	14384 (2.9)	24844 (7.0)	22.00 ± 6.782
Gaschl	14.00 ± 5.773	18.50 ± 6.819	19.20 ± 7.814	30	20070 (3.4)	37838 (9.2)	26.70 ± 5.908
Gasch2	18.10 ± 8.800	20.40 ± 7.501	23.70 ± 5.437	27	18546 (3.2)	34204 (8.3)	39.20 ± 8.741
Pheno	5.40 ± 1.776	5.40 ± 1.350	4.90 ± 1.287	6	5691 (1.3)	6213 (2.0)	5.30 ± 0.483
Spo	13.30 ± 5.677	12.40 ± 4.624	12.00 ± 4.59	14	15552 (2.7)	35400 (8.6)	29.40 ± 7.152
Expr	14.60 ± 5.910	16.50 ± 4.453	18.00 ± 5.375	35	20812 (3.6)	38313 (9.3)	29.80 ± 4.289
Seq	13.00 ± 5.518	11.70 ± 5.478	15.80 ± 7.162	15	21703 (3.7)	38969 (9.4)	15.90 ± 3.784
Média	13.16	15.46	16.66	20.2	16490.4 (2.91)	29926.5 (15.69)	26.64

Como pode ser observado nas Tabelas 6.22 e 6.23, o número e tamanho das regras geradas pelo método **HMC-GA** foi bastante competitivo com o número e tamanho das regras geradas pelos métodos da literatura. Se for observado, por exemplo, o conjunto de dados Eisen estruturado como grafo, tem-se que o método **HMC-GA** obteve, em média, um número de regras igual à metade do número de regras obtidas pelo método *hmAnt-Miner*, e quase três vezes menos regras do que o método Clus-HMC. O tamanho das regras geradas também foi, em média, bem competitivo com o tamanho das regras geradas pelo método *hmAnt-Miner*, e igual à metade do tamanho das regras geradas pelo método Clus-HMC. Esse bom desempenho, considerando a quantidade e tamanho das regras, foi acompanhado por um desempenho preditivo competitivo. O método **HMC-GA** obteve o mesmo desempenho que o método Clus-HMC, e junto com esse, superou todos os outros métodos considerando a medida $AU(\overline{PRC})$. Ainda, pode-se observar

Tabela 6.23: Tamanho das regras geradas pelos métodos HMC-GA, Clus-HMC e *hmAnt-Miner*

Dados	100 Indivíduos	HMC-GA 500 Indivíduos	1000 Indivíduos	Clus-HMC	<i>hmAnt-Miner</i>
Conjuntos de dados estruturados como árvore					
Celcycle	3.99 ± 0.451	4.71 ± 0.470	5.14 ± 0.413	5.29	3.67 ± 0.336
Church	5.89 ± 0.605	6.56 ± 0.508	6.82 ± 0.535	4.76	3.12 ± 0.427
Derisi	5.54 ± 1.249	5.99 ± 1.114	6.48 ± 0.669	1.75	5.21 ± 0.631
Eisen	3.39 ± 0.725	4.74 ± 0.414	5.43 ± 0.923	5.69	3.54 ± 0.341
Gasch1	4.27 ± 0.636	5.31 ± 0.730	5.70 ± 0.526	3.30	3.75 ± 0.381
Gasch2	3.95 ± 0.622	5.05 ± 0.509	5.21 ± 0.389	5.77	3.91 ± 1.274
Pheno	12.31 ± 2.115	11.26 ± 2.198	13.17 ± 1.049	3.12	3.22 ± 0.632
Spo	6.01 ± 1.400	7.48 ± 1.231	7.58 ± 0.868	2.17	4.51 ± 0.442
Expr	4.18 ± 1.044	5.72 ± 1.392	6.62 ± 0.992	3.83	3.54 ± 0.270
Seq	5.48 ± 0.884	6.46 ± 1.834	6.32 ± 1.733	4.21	3.22 ± 0.382
Média	5.50	6.33	6.85	3.99	3.77
Conjuntos de dados estruturados como grafo					
Celcycle	4.03 ± 0.440	4.95 ± 0.498	5.14 ± 0.553	6.29	4.13 ± 0.289
Church	5.61 ± 0.637	6.27 ± 0.309	6.62 ± 0.660	3.00	3.98 ± 0.248
Derisi	4.95 ± 0.750	6.29 ± 0.930	6.31 ± 0.679	3.30	5.29 ± 0.502
Eisen	3.12 ± 0.952	4.61 ± 0.374	5.38 ± 0.554	6.12	3.77 ± 0.430
Gasch1	3.99 ± 0.966	5.44 ± 0.683	5.85 ± 0.459	5.70	3.38 ± 0.418
Gasch2	3.91 ± 0.717	4.90 ± 0.390	5.44 ± 0.805	5.63	4.18 ± 0.280
Pheno	16.43 ± 3.56	16.41 ± 2.758	15.41 ± 2.639	3.33	9.31 ± 1.618
Spo	6.10 ± 1.349	7.58 ± 1.052	9.10 ± 0.855	4.29	5.82 ± 0.365
Expr	4.56 ± 1.183	6.36 ± 0.632	6.91 ± 0.915	5.94	3.25 ± 0.188
Seq	4.34 ± 0.561	6.03 ± 1.018	6.80 ± 0.838	4.60	3.06 ± 0.289
Média	5.70	6.88	7.30	4.82	4.62

nas tabelas que, na maioria dos casos, houve um aumento no número de regras e no tamanho das regras conforme o número de indivíduos da população foi aumentado.

Também é importante observar nas Tabelas 6.22 e 6.23 a grande diferença entre o número de regras geradas pelos métodos locais e pelos métodos globais. Como esperado, os métodos locais geraram uma quantidade muito maior de regras. Sendo assim, o modelo induzido é muito menos interpretável do que o conjunto de regras geradas pelos métodos globais. Ainda, os tamanhos das regras geradas pelos métodos locais não são apresentados porque o formato da saída gerada pelos métodos Clus-HSC e Clus-SC não permitiu que essa informação fosse extraída. De qualquer maneira, um conjunto muito grande de regras pequenas é menos interpretável do que um conjunto pequeno de regras também razoavelmente pequenas.

A Tabela 6.24 mostra os rankings médios do método HMC-GA utilizado com tamanhos de população igual a 100, 500 e 1000 indivíduos, e também dos métodos da literatura, de acordo com o teste de Friedman, considerando separadamente a quantidade de regras geradas e o tamanho das regras geradas. Após a aplicação do teste, os valores de p -value obtidos considerando quantidade e tamanho das regras foram, respectivamente, 9.21×10^{-34} e 5.40×10^{-11} , o que indica claramente que houve diferenças estatisticamente significantes entre os métodos. Para identificar as comparações par a par nas quais há diferenças estatisticamente significantes, o teste post-hoc de Nemenyi foi aplicado. Os diagramas da Figura 6.6 mostram os resultados do teste de Nemenyi considerando todos os pares de comparações. São conectados os métodos em cujas comparações não foram detectadas diferenças estatisticamente significantes.

De acordo com a Figura 6.6, não houve diferenças estatisticamente significantes na comparação das quantidades de regras geradas pelos métodos HMC-GA e Clus-HMC. No entanto, na

Tabela 6.24: Rankings médios do método HMC-GA e métodos da literatura de acordo com o teste de Friedman, considerando quantidade e tamanho das regras geradas.

Método	Ranking da quantidade	Método	Ranking do tamanho
HMC-GA (100 indivíduos)	1.77	<i>hmAnt-Miner</i>	1.65
HMC-GA (500 indivíduos)	2.72	HMC-GA (100 indivíduos)	2.45
HMC-GA (1000 indivíduos)	2.90	Clus-HMC	2.70
Clus-HMC	3.25	HMC-GA (500 indivíduos)	3.65
<i>hmAnt-Miner</i>	4.35	HMC-GA (1000 indivíduos)	4.55
Clus-HSC	6.00	-	-
Clus-SC	7.00	-	-

Figura 6.6: Comparação para a par do método HMC-GA e métodos da literatura utilizando o teste estatístico de Nemenyi, considerando a quantidade e tamanho das regras geradas. (a) resultados considerando a quantidade de regras; (b) resultados considerando o tamanho das regras.

comparação com todos os outros métodos, o algoritmo genético utilizando 100 indivíduos obteve resultados estatisticamente superiores. Ainda, no diagrama ilustrado no item (a) da figura, tem-se que o método *hmAnt-Miner* não foi estatisticamente superior ao método Clus-HSC, apesar desse último ter gerado um número muito maior de regras do que o primeiro. Isso aconteceu porque o teste considera os rankings dos métodos, e não os valores absolutos. Se fosse realizado um teste não paramétrico considerando esses dois métodos, o resultado apontaria diferença estatisticamente significativa, já que o método *hmAnt-Miner* obteve menos regras que o método Clus-HSC em todos os conjuntos de dados.

Considerando os tamanhos das regras geradas (Figura 6.6(b)), não houve diferenças estatisticamente significantes na comparação dos métodos HMC-GA com 100 indivíduos, *hmAnt-Miner* e Clus-HMC. Observa-se que conforme o número de indivíduos foi aumentado, diferenças estatisticamente significantes foram encontradas quando comparando o método HMC-GA com os outros métodos. Com 500 indivíduos, o algoritmo genético perdeu estatisticamente apenas para o método *hmAnt-Miner*, enquanto que com a utilização de 1000 indivíduos, o algoritmo genético foi estatisticamente inferior também ao método Clus-HMC.

Os resultados experimentais para o método HMC-GA são bastante animadores quanto à quantidade e tamanho médio das regras geradas em execuções individuais. Observou-se que, em muitos casos, as execuções que levaram aos melhores resultados preditivos foram as execuções que resultaram na indução do menor número de regras. Ainda, em algumas dessas

execuções, os resultados obtidos foram competitivos ou melhores se comparado com os resultados obtidos pelos métodos da literatura. A Tabela 6.25 faz um comparativo entre os valores de $AU(\overline{PRC})$ obtidos nessas execuções e os resultados obtidos pelos outros métodos. São apresentados os resultados obtidos pelo algoritmo genético com a população de tamanho igual a 1000 indivíduos, já que com esse número de indivíduos o método **HMC-GA** obteve resultados um pouco melhores, ainda mantendo um tamanho e número de regras similar à utilização de 100 e 500 indivíduos. São mostrados na tabela os valores de $AU(\overline{PRC})$, e entre parênteses a quantidade/tamanho médio das regras geradas. Novamente não são mostrados os tamanhos das regras geradas pelos métodos Clus-HSC e Clus-SC, pois os mesmos não fornecem as regras como saída. Os valores nos quais o método **HMC-GA** obteve melhores resultados são destacados em negrito.

Tabela 6.25: Valores de $AU(\overline{PRC})$ e número/tamanho de regras obtidos em execuções específicas do método **HMC-GA**.

Conjuntos de Dados	HMC-GA	Clus-HMC	Clus-HSC	Clus-SC	<i>hmAnt-Miner</i>
Conjuntos de dados estruturados como árvore					
Celcycle	0.175 (8/6)	0.172 (24/5.29)	0.111 (4037)	0.106 (9671)	0.155 (25.10/3.67)
Church	0.166 (8/6.25)	0.170 (17/4.76)	0.131 (2221)	0.128 (4186)	0.165 (8.10/3.12)
Derisi	0.168 (6/6.33)	0.175 (4/1.75)	0.094 (3520)	0.089 (7807)	0.149 (26.60/5.21)
Eisen	0.210 (11/6.54)	0.204 (29/5.69)	0.127 (2995)	0.132 (6311)	0.181 (17.50/3.54)
Gasch1	0.200 (8/6)	0.205 (10/3.30)	0.106 (4761)	0.104 (10447)	0.173 (28.10/3.75)
Gasch2	0.178 (10/5.80)	0.195 (26/5.77)	0.121 (3756)	0.119 (7850)	0.152 (36.90/3.91)
Pheno	0.161 (3/15)	0.160 (8/3.12)	0.152 (777)	0.149 (1238)	0.161 (4.90/3.22)
Spo	0.182 (9/8.11)	0.186 (6/2.17)	0.103 (3623)	0.098 (8527)	0.177 (12.80/4.51)
Expr	0.198 (11/8.54)	0.210 (12/3.83)	0.127 (4711)	0.123 (10262)	0.180 (28.60/3.54)
Seq	0.166 (3/3.33)	0.211 (14/4.21)	0.091 (4923)	0.095 (10443)	0.186 (15.30/3.22)
Conjuntos de dados estruturados como grafo					
Celcycle	0.348 (18/5.78)	0.357 (21/6.29)	0.371 (19085)	0.252 (36260)	0.325 (39.30/4.13)
Church	0.350 (4/6.25)	0.348 (7/3.0)	0.397 (12368)	0.289 (16049)	0.334 (25.90/3.98)
Derisi	0.350 (8/5.87)	0.355 (10/3.30)	0.349 (16693)	0.218 (31175)	0.321 (32.90/5.29)
Eisen	0.389 (6/6.67)	0.380 (37/6.12)	0.365 (14384)	0.270 (24844)	0.373 (22.0/3.77)
Gasch1	0.365 (8/5.87)	0.371 (30/5.70)	0.351 (20070)	0.239 (37838)	0.352 (26.70/3.38)
Gasch2	0.355 (11/7.45)	0.365 (27/5.63)	0.378 (18546)	0.267 (34204)	0.334 (39.20/4.18)
Pheno	0.335 (2/10.50)	0.337 (6/3.33)	0.416 (5691)	0.316 (6213)	0.336 (5.30/9.31)
Spo	0.352 (5/10.20)	0.352 (14/4.29)	0.371 (15552)	0.213 (35400)	0.326 (29.40/5.82)
Expr	0.374 (13/7.08)	0.368 (35/5.94)	0.351 (20812)	0.249 (38313)	0.343 (29.80/3.25)
Seq	0.379 (10/7.90)	0.386 (15/4.60)	0.282 (21703)	0.197(38969)	0.371 (15.90/3.06)

Apesar de os resultados apresentados na Tabela 6.25 não serem válidos para dizer que o método **HMC-GA** obteve resultados melhores que o método estado-da-arte na literatura, eles servem para reforçar a tese de que o algoritmo genético proposto é capaz de gerar regras pequenas e em menor número do que os métodos da literatura, e ainda assim obter desempenhos de classificação melhores que esses métodos. Aperfeiçoamentos no método podem levar a uma maior estabilidade do algoritmo, fazendo com que bons resultados sejam obtidos em várias execuções, tornando o método mais consistente.

A Figura 6.7 apresenta exemplos de regras geradas pelo método **HMC-GA**, com 1000 indivíduos, no conjunto de dados Eisen estruturado tanto como árvore quanto grafo. São apresentadas as regras geradas nas execuções que resultaram na indução do menor número de regras. São mostrados também exemplos de regras geradas pelos métodos Clus-HMC e *hmAnt-Miner*, apenas para ilustrar as regras geradas pelos diferentes métodos. São apresentadas, para cada es-

trutura hierárquica, a maior regra do conjunto de regras gerado por cada método. Para o método *hmAnt-Miner*, as regras apresentadas foram geradas nas execuções que resultaram nos maiores valores de $AU(\overline{PRC})$ no conjunto de treinamento.

HMC-GA:	Clus-HMC:	hmAnt-Miner:
<pre> SE alpha_91 >= -0.64 E alpha_119 >= -0.76 E elu_90 >= -0.61 E -0.67 <= cdc15_130 <= 1.98 E spo_2 <= 0.069 E spo_5 <= -0.065 E spo_11 >= -1.93 E spo5_11 >= -0.56 E heat_20 <= -0.92 E dtt_120 <= -0.84 E diau_780 <= 0.77 ENTÃO ... </pre>	<pre> SE heat_20 > -1.18 E diau_1260 <= 1.14 E spo_mid <= 1.06 E alpha_0 <= 1.24 E dtt_60 <= -0.04 E spo_early <= -0.51 E cdc15_250 > 0.68 E diau_900 > -0.34 E elu_120 > -0.3 ENTÃO ... </pre>	<pre> SE alpha_56 < -0.05 E dtt_60 >= -0.28 E heat_0 < -0.135 E elu_240 >= -0.2 E cdc15_210 >= -0.57 E spo_2 >= 0.235 E dtt_30 >= -0.635 ENTÃO ... </pre>
(a)		
HMC-GA:	Clus-HMC:	hmAnt-Miner:
<pre> SE alpha_0 >= -0.69 E alpha_28 <= 0.34 E elu_270 >= -0.50 E elu_330 >= -1.41 E spo_2 <= -0.048 E -6.35 <= spo_7 <= -0.23 E spo5_11 >= 0.028 E heat_20 <= -1.018 E heat_40 <= -0.74 E dtt_120 <= -0.76 E cold_160 <= -0.57 E diau_1020 <= 0.84 E diau_1260 <= -0.71 ENTÃO ... </pre>	<pre> SE heat_20 > -2.0 E heat_10 <= 0.3 E spo_early <= 0.77 E cdc15_290 <= 0.81 E heat_20 <= -0.74 E elu_330 > -0.49 E spo5_11 > 0.52 E dtt_120 <= -0.84 E heat_40 > -1.43 ENTÃO ... </pre>	<pre> SE dtt_120 < -0.825 E cold_160 < -0.905 E alpha_21 < 0.795 E diau_1260 < -0.85 E heat_40 < -0.665 E alpha_98 >= -0.185 E diau_900 >= -0.61 ENTÃO ... </pre>
(b)		

Figura 6.7: Regras geradas pelos métodos HMC-GA, Clus-HMC e *hmAnt-Miner*. (a) conjunto de dados Eisen estruturado como árvore; (b) conjunto de dados Eisen estruturado como grafo.

Como pode ser observado na Figura 6.7, o algoritmo genético com 1000 indivíduos gerou regras um pouco maiores do que as regras geradas pelos outros métodos. Isso é condizente com os resultados apresentados na Tabela 6.23, que mostra que em média, o método *HMC-GA* gerou regras um pouco maiores que as dos outros métodos. Ainda que em média as regras geradas pelo algoritmo genético sejam um pouco maiores, pode-se dizer que os conjuntos de regras do método *HMC-GA* são tão simples quanto, ou mais simples, que os conjuntos de regras gerados pelos outros métodos, já que o algoritmo genético gerou bem menos regras.

6.6 Gráficos das curvas precisão-revocação

As Figuras 6.8 e 6.9 apresentam os gráficos das curvas precisão-revocação obtidas por todos os métodos, respectivamente nos conjuntos de dados estruturados como árvores e como grafos. Para o método *HMC-LMLP*, são apresentadas as curvas *PR* referentes aos resultados obtidos pela variação que utiliza as predições para complementar os vetores de atributos dos exemplos (*HMC-LMLP-Predicted*). São mostrados os resultados obtidos com os algoritmos *Back-propagation* (*Bp-Predicted*) e *Resilient Back-propagation* (*Rp-Predicted*). Ainda, as curvas apresentadas são aquelas obtidas nas execuções do método que resultaram nos melhores valores de $AU(\overline{PRC})$ nos conjuntos de validação.

Como o objetivo do método **HMC-GA** é gerar um conjunto pequeno de regras mantendo o bom desempenho de classificação, as curvas apresentadas são aquelas obtidas nas execuções do método que resultaram na obtenção do menor número de regras. São mostrados os resultados obtidos com a utilização de 1000 indivíduos no algoritmo genético. Para o método *hmAnt-Miner*, são mostradas as curvas obtidas nas execuções que resultaram nos melhores valores de $AU(\overline{PRC})$ nos conjuntos de treino. Como os métodos Clus-HMC, Clus-HSC e Clus-SC são determinísticos, apenas uma única execução de cada um deles é necessária. Assim, os gráficos das curvas **PR** obtidas nessas execuções são apresentados.

Como pode ser observado na Figura 6.8, as comparações entre os resultados nos conjuntos de dados estruturados como árvores mostram que as curvas **PR** obtidas pelo método **HMC-LMLP** ficam, em alguns trechos, acima das curvas **PR** obtidas pelos outros métodos. Essa diferença de desempenho é mais evidente nos conjuntos de dados Cellcycle (Figura 6.8(a)), Derisi (Figura 6.8(c)), Eisen (Figura 6.8(d)), Gasch1 (Figura 6.8(e)), Gasch2 (Figura 6.8(f)), Expr (Figura 6.8(i)) e Seq (Figura 6.8(j)).

Ainda nas hierarquias estruturadas como árvores, pode-se observar pelas curvas do método **HMC-GA** que o mesmo obteve, em alguns casos, resultados competitivos com o método Clus-HMC. Tal desempenho pode ser observado especialmente nos conjuntos de dados Cellcycle (Figura 6.8(a)), Derisi (Figura 6.8(c)), Eisen (Figura 6.8(d)), Gasch1 (Figura 6.8(e)) e Expr (Figura 6.8(i)). Tal resultado pode ser considerado animador, dado que o método **HMC-GA** obteve, em alguns desses conjuntos de dados, um número de regras menor que o número de regras obtidas pelo método Clus-HMC. Ainda, é possível observar nos gráficos como o algoritmo genético superou as versões locais dos métodos baseados em **PCT**, Clus-HSC e Clus-SC.

Nas curvas **PR** obtidas nos conjuntos de dados estruturados como grafos (Figura 6.9), tem-se um maior equilíbrio entre os desempenhos dos métodos, o que pode ser observado por meio da sobreposição das curvas **PR**. Percebe-se nos gráficos apresentados, que os métodos **HMC-LMLP** e **HMC-GA** foram bem competitivos com os métodos da literatura. Observa-se ainda como o método Clus-SC obteve desempenhos muito inferiores aos desempenhos dos outros métodos, o que foi causado pelo fato desse método não considerar nenhum relacionamento hierárquico entre as classes. O método já havia obtido um desempenho ruim nos conjuntos de dados estruturados como árvores. Assim, nos conjuntos de dados estruturados como grafos, que possuem muito mais relacionamentos hierárquicos, sua desvantagem ficou mais evidente.

Ainda nos conjuntos de dados estruturados como grafos, pode-se observar que as áreas abaixo das curvas **PR** obtidas são muito maiores do que as áreas das curvas **PR** obtidas nos conjuntos de dados estruturados como árvores. Isso pode ser explicado pelo fato de que apesar de possuir um número muito maior de classes, os conjuntos de dados estruturados como grafos possuem muito mais exemplos positivos para cada classe. Isso pode ser observado na Tabela 6.4, que mostra o número total de exemplos e o número de exemplos multirrótulo dos conjuntos de dados. Tem-se que, diferentemente das hierarquias estruturadas como árvores, todos os exemplos dos conjuntos de dados estruturados como grafos são classificados em mais de uma classe simultaneamente.

Figura 6.8: Curvas precisão-revocação obtidas nas hierarquias estruturadas como árvores

Figura 6.9: Curvas precisão-revocação obtidas nas hierarquias estruturadas como grafos

6.7 Considerações Finais

Neste capítulo foram apresentados os experimentos realizados com os métodos *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons* (HMC-LMLP) e *Hierarchical Multi-Label Classification with a Genetic Algorithm* (HMC-GA). Como o método HMC-LMLP é baseado na abordagem local, foi discutido o desempenho do método nível a nível nas hierarquias, e em classes específicas em cada nível. Foi mostrado também como as redes neurais obtiveram resultados superiores a todos os outros métodos nos conjuntos de dados estruturados como árvores. Nos conjuntos de dados estruturados como grafos, foi feita uma adaptação nas hierarquias para definir a profundidade das classes, e permitir a aplicação do método HMC-LMLP. O método obteve resultados competitivos com os outros métodos da literatura, e poderia ter obtido resultados melhores, não fossem as desvantagens discutidas decorrentes da adaptação feita nas hierarquias.

Foi mostrado também como, em alguns casos, o método HMC-GA obteve resultados de classificação competitivos com os outros métodos da literatura. Ainda, apesar de não superar o método estado-da-arte na literatura, foi mostrado como, em muitos casos, as regras geradas pelo algoritmo genético foram menores e em menor número do que as regras geradas pelos outros métodos. Ainda, em algumas execuções específicas, as regras geradas foram em menor número e obtiveram desempenhos de classificação superiores aos métodos da literatura.

Os gráficos das curvas precisão-revocação de todos os métodos também foram apresentados, mostrando como os métodos HMC-LMLP e HMC-GA, principalmente as redes neurais, obtiveram desempenhos competitivos com os desempenhos dos métodos da literatura. A próxima seção apresenta as principais conclusões desta Tese e futuras pesquisas.

Conclusões

Nesta Tese de Doutorado foram propostos e investigados dois métodos de classificação para problemas hierárquicos multirrótulo, chamados [HMC-LMLP](#) e [HMC-GA](#). Estes métodos utilizam, respectivamente, redes neurais artificiais do tipo *Multi-Layer Perceptron* ([MLP](#)) e Algoritmos Genéticos ([AG](#)). O método [HMC-LMLP](#) tem como prioridade apenas o desempenho preditivo em tarefas de classificação, enquanto o método [HMC-GA](#) tem como objetivo também gerar regras de classificação hierárquicas multirrótulo para facilitar a interpretação dos modelos induzidos.

Em problemas de classificação hierárquica multirrótulo, exemplos são classificados em classes estruturadas em uma hierarquia, sendo que um exemplo pode simultaneamente ser classificado em mais de uma classe em cada nível, ou seja, exemplos podem ser classificados em diferentes caminhos da hierarquia ao mesmo tempo. Além disso, as classes do problema podem ser estruturadas em uma hierarquia do tipo árvore ou do tipo Grafo Acíclico Direcionado ([GAD](#)).

A principal diferença entre as hierarquias estruturadas como árvores e grafos está no número de pais que cada classe pode ter. Em hierarquias estruturadas como árvores, cada classe pode possuir apenas uma classe pai, enquanto em hierarquias estruturadas como grafos, uma classe pode possuir mais de um pai. Essa diferença tem influência na profundidade das classes na hierarquia. Definindo a profundidade de uma classe como o número de arestas do caminho entre a raiz e a classe, em hierarquias do tipo árvore, uma classe possui apenas uma profundidade, já que existe apenas um caminho possível entre o nó raiz da hierarquia e a classe. Já em hierarquias estruturadas como grafos, a profundidade de uma classe depende do caminho escolhido, pois caminhos diferentes podem ser percorridos entre o nó raiz e uma classe na hierarquia.

Duas abordagens são utilizadas para tratar problemas hierárquicos multirrótulo, chamadas global e local. Na abordagem global, um único classificador é induzido considerando todas as classes da hierarquia ao mesmo tempo. Assim, a classificação de um novo exemplo acontece em apenas um passo. Na abordagem local, ao contrário, um conjunto de classificadores é induzido,

sendo cada classificador responsável pela predição de uma classe, ou de um conjunto de classes. A classificação de novos exemplos então ocorre em vários passos, combinando as predições dos vários classificadores previamente treinados.

Os experimentos desta Tese foram realizados com a finalidade de comparar os desempenhos de classificação dos métodos propostos — [HMC-LMLP](#) e [HMC-GA](#) — com os desempenhos de classificação dos métodos da literatura. Para isso, foram escolhidos métodos baseados tanto na abordagem global quanto local. Baseados na abordagem global, foram utilizados os métodos Clus-HMC e *hmAnt-Miner*. O primeiro induz uma única árvore de decisão para lidar com todas as classes da hierarquia, e utiliza o conceito de *Predictive Clustering Trees* ([PCT](#)), no qual uma árvore de decisão é vista como uma hierarquia de grupos (*clusters*), e o segundo utiliza *Ant Colony Optimization* ([ACO](#)) para a indução de regras de classificação hierárquicas multirrótulo, ou seja, uma regra deve ser capaz de classificar um exemplo em um conjunto de classes, respeitando as restrições impostas pela hierarquia.

Baseados na abordagem local, foram utilizados nos experimentos os métodos Clus-HSC e Clus-SC. Assim como o método Clus-HMC, ambos utilizam o conceito de [PCT](#). O método Clus-HSC induz um conjunto de árvores de decisão, associando cada uma delas a uma classe da hierarquia, utilizando os relacionamentos hierárquicos entre as classes durante o processo de indução. O método Clus-SC também induz um classificador para cada classe da hierarquia, porém ignorando os relacionamentos hierárquicos entre as classes.

Além de utilizar métodos locais e globais, os experimentos envolveram o uso de hierarquias estruturadas como grafos e árvores. Foram utilizados conjuntos de dados relacionados à tarefa de predição de funções de proteínas, estruturados de acordo com o esquema de classificação FunCat (árvore), e também de acordo com a *Gene Ontology* ([GO](#)) ([DAG](#)). Todos os conjuntos de dados utilizados estão disponíveis em <http://www.cs.kuleuven.be/~dtai/clus/hmcdatasets.html>, e já foram utilizados por outros trabalhos da literatura.

Os resultados dos experimentos foram analisados utilizando curvas precisão-revocação. As medidas de precisão e revocação utilizadas foram desenvolvidas especificamente para problemas de classificação hierárquica multirrótulo, e foram escolhidas por apresentarem vantagens em comparação com outras medidas da literatura. Entre essas vantagens, estão a possibilidade de utilização direta na avaliação de classificadores aplicados a qualquer tipo de taxonomia hierárquica, e também a independência da utilização de parâmetros.

Os resultados dos experimentos foram reportados separadamente para cada um dos dois métodos propostos. Como o método [HMC-LMLP](#) é baseado na abordagem local, foram analisados os resultados nível a nível das hierarquias, e também resultados em classes específicas de alguns desses níveis. Já para o método [HMC-GA](#), optou-se apenas pela avaliação do desempenho geral do método, e também pela avaliação das regras induzidas. Isso foi feito pelo fato do método ser baseado na abordagem global de classificação, que tem como objetivo fazer a classificação considerando todas as classes ao mesmo tempo. Assim, acredita-se não fazer sentido uma análise nível a nível das hierarquias.

O restante deste Capítulo está organizado da seguinte maneira. A Seção [7.1](#) destaca as

contribuições desta pesquisa de Doutorado; na Seção 7.2 são apresentados resumidamente os principais resultados obtidos; por fim, a Seção 7.3 apresenta algumas pesquisas futuras que pretendem ser realizadas.

7.1 Contribuições da Pesquisa

A principal contribuição desta pesquisa de Doutorado foi a proposta e investigação de dois métodos de classificação hierárquica multirrótulo, chamados *Hierarchical Multi-Label Classification with Local Multi-Layer Perceptrons (HMC-LMLP)* e *Hierarchical Multi-Label Classification with a Genetic Algorithm (HMC-GA)*.

O método **HMC-LMLP** é baseado na abordagem local de classificação, utilizando a estratégia *Local Classifier per Level (LCL)*. O método associa uma rede neural a cada nível hierárquico, sendo cada uma responsável pelas predições em seu nível correspondente. Não se tem conhecimento na literatura de nenhum outro trabalho que utilize a estratégia **LCL** para a tarefa de classificação hierárquica multirrótulo.

Quatro variações do método **HMC-LMLP** foram propostas. Na primeira, redes neurais são treinadas incrementalmente, utilizando as predições da **MLP** associada ao nível $l - 1$ para complementar os vetores de atributos dos exemplos de treinamento do nível l . Uma segunda variação utiliza as classes verdadeiras dos exemplos para essa complementação. A terceira variação não utiliza nenhuma informação de classes para complementar os vetores de atributos dos exemplos, e a última variação utiliza, a partir do segundo nível, apenas as predições no nível $l - 1$ como entrada da **MLP** associada ao nível l . O objetivo dos experimentos foi mostrar que a utilização das predições levou a melhores resultados, permitindo que a rede **MLP** associada ao nível l utilizasse informações aprendidas previamente pela **MLP** associada ao nível $l - 1$, e que os relacionamentos entre as classes foram aprendidos pelas redes neurais durante o processo de indução. Os experimentos com o método **HMC-LMLP** resultaram nas seguintes publicações:

Periódicos internacionais

- Cerri, R., Barros, R. C., Carvalho, A. C. P. L. F.: **Hierarchical Multi-label Classification using Local Neural Networks**. In: Journal of Computer and System Sciences, volume 80, p. 39-56, 2014.

Conferências

- Cerri, R., Barros, R. C., Carvalho, A. C. P. L. F.: **Hierarchical Multi-label Classification for Protein Function Prediction: A Local Approach based on Neural Networks**. In: 11th International Conference on Intelligent Systems Design and Applications, Córdoba, Espanha, p. 337-343, 2011.
- Cerri, R., Barros, R. C., Carvalho, A. C. P. L. F.: **Redes Neurais Artificiais Aplicadas a Problemas de Classificação Hierárquica Multirrótulo**. In: Congresso Brasileiro de

Inteligência Computacional, Fortaleza. Anais do Congresso Brasileiro de Inteligência Computacional. Rio de Janeiro: SBIC, p. 1-8, 2011.

- Cerri, R., Carvalho, A. C. P. L. F.: **Hierarchical Multilabel Protein Function Prediction Using Local Neural Networks.** In: Brazilian Symposium on Bioinformatics, Brasília - DF. Advances in Bioinformatics and Computational Biology. Berlin: Springer, v. 6832. p. 10-17, 2011

Ainda, um artigo discutindo todas as quatro variações do método [HMC-LMLP](#) acaba de ser submetido a um periódico internacional. Trata-se do seguinte artigo:

- Cerri, R., Barros, R. C., Carvalho, A. C. P. L. F., Jin, Y.: **Neural Networks for Hierarchical Multi-Label Classification.** Submetido ao periódico IEEE Transactions on Neural Networks and Learning Systems.

O método [HMC-GA](#) é baseado na abordagem global de classificação, e utiliza um algoritmo genético para induzir regras de classificação que sejam tanto hierárquicas quanto multirrótulo. Até o momento, tem-se conhecimento de apenas um trabalho que utilize algoritmos genéticos para a tarefa de classificação hierárquica. No entanto, o método é baseado na abordagem *Local Classifier per Node* ([LCN](#)), e foi aplicado a problemas hierárquicos simples-rótulo.

As regras do método [HMC-GA](#) são induzidas utilizando um procedimento de cobertura sequencial de exemplos. Nesse procedimento, ao final de um processo evolutivo, um conjunto de regras é induzido e a melhor regra é salva. Os exemplos cobertos pela melhor regra são então removidos do conjunto de treinamento, e um novo processo evolutivo é iniciado. Isso é repetido até que todos, ou quase todos, os exemplos de treinamento sejam cobertos. Ainda, no momento de cruzar duas regras para dar origem a regras filhas, o método utiliza um procedimento de seleção que considera a distância entre duas regras candidatas. Esse procedimento calcula a Distância Euclidiana entre os consequentes de duas regras (vetor médio de classes dos exemplos cobertos pela regra). Quanto menor a Distância Euclidiana, mais próximas as regras são consideradas. A ideia é cruzar regras que cubram exemplos próximos no espaço de busca, ou seja, exemplos classificados em conjuntos de classes parecidos. Os experimentos preliminares com o método [HMC-GA](#) resultaram na seguinte publicação:

- Cerri, R., Barros, R. C., Carvalho, A. C. P. L. F.: **A Genetic Algorithm for Hierarchical Multi-Label Classification.** In: 27th ACM Symposium On Applied Computing, Riva del Garda, Trento. Proceedings of the 27th ACM Symposium On Applied Computing, p. 250-255, 2012

Além dessa publicação, um artigo descrevendo os experimentos realizados nesta Tese com o método [HMC-GA](#) está sendo redigido para ser submetido a um periódico internacional.

A aplicação do método [HMC-LMLP](#) nos conjuntos de dados estruturados como [DAGs](#) também pode ser considerada uma contribuição desta pesquisa. Por meio da adaptação realizada

nas hierarquias, definindo os níveis hierárquicos de suas classes, foi possível aplicar a estratégia local **LCL**. Ainda, verificou-se que essa adaptação prejudicou o desempenho do método **HMC-LMLP**, pois nem todos os relacionamentos hierárquicos entre as classes puderam ser utilizados quando os vetores de atributos de um exemplo do nível l eram complementados com as predições obtidas para o exemplo no nível $l - 1$.

Além dos métodos **HMC-LMLP** e **HMC-GA**, a revisão bibliográfica de métodos e medidas de avaliação para problemas hierárquicos multirrótulo também pode ser considerada uma contribuição desta pesquisa. Experimentos comparando diferentes métodos de classificação e também diferentes medidas de avaliação resultaram na seguinte publicação em um periódico internacional:

- Cerri, R., Pappa, G. L., Carvalho, A. C. P. L. F., Freitas, A. A.: **An Extensive Evaluation of Decision Tree-based Hierarchical Multilabel Classification Methods and Performance Measures**. In: Computational Intelligence, Disponível online: 30 de Maio de 2013, ISSN 1467-8640, <http://dx.doi.org/10.1111/coin.12011> 2013. (Aceito para publicação)

7.2 Principais Resultados

Como um dos principais resultados desta pesquisa, pode-se destacar o desempenho de classificação obtido pelo método **HMC-LMLP**. Nos conjuntos de dados estruturados como árvores, o método obteve desempenho superior a todos os outros métodos da literatura na grande maioria dos conjuntos de dados. Ainda, os resultados nível a nível e em classes específicas mostraram que o método foi capaz de superar os outros métodos tanto nas classes mais frequentes quanto nas classes menos frequentes das hierarquias.

Nos conjuntos de dados estruturados como grafos, o método **HMC-LMLP** mostrou-se bastante competitivo na comparação com os outros métodos da literatura. Principalmente quando utilizando o algoritmo de treinamento *Resilient Back-propagation*, as redes neurais obtiveram, em alguns conjuntos de dados, desempenhos superiores ao método Clus-HMC, considerado estado-da-arte, e ao método Clus-HSC. Ainda, foram obtidos desempenhos superiores na comparação com os métodos Clus-SC e *hmAnt-Miner*.

Considerando as avaliações nível a nível e em classes específicas das hierarquias estruturadas como grafos, foi possível observar que o método **HMC-LMLP** obteve, inclusive, resultados superiores aos métodos da literatura em classes pouco frequentes, o que explica o pior desempenho do método na medida de avaliação $\overline{AUPRC}_w$.

Ainda nos experimentos com o método **HMC-LMLP**, acredita-se que o mesmo não obteve, nos conjuntos de dados estruturados como grafos, desempenhos tão bons quanto nos conjuntos de dados estruturados como árvores, devido à adaptação realizada nas hierarquias. Como discutido nos experimentos, a definição do nível hierárquico eliminou informações relacionadas a todas as classes pais de uma classe. Assim, durante o processo de treinamento, não puderam

ser utilizadas todas as predições feitas para as classes pais de uma dada classe no momento da complementação dos vetores de atributos dos exemplos.

Nos experimentos com o método **HMC-GA**, foi observado que o método obteve resultados competitivos na comparação com a maioria dos métodos da literatura, porém foi superado pelo método estado-da-arte (Clus-HMC). Considerando a medida $AU(\overline{PRC})$, foram obtidos desempenhos competitivos nos conjuntos de dados estruturados de acordo com ambas as hierarquias em árvore e grafo. No entanto, na medida $\overline{AUPRC}_w$, o desempenho do método **HMC-GA** pode ser considerado competitivo com alguns dos métodos da literatura apenas nas hierarquias estruturadas como árvores. Nas hierarquias estruturadas como grafos, os valores de $\overline{AUPRC}_w$ mostram que os métodos da literatura foram melhores que o algoritmo genético nas classes mais frequentes dos conjuntos de dados, e que o método **HMC-GA** foi melhor ou mais competitivo nas classes menos frequentes.

Apesar de o método **HMC-GA** não ter superado o método estado-da-arte na literatura, os resultados obtidos podem ser considerados animadores, já que, em média, o algoritmo genético gerou um conjunto menor de regras do que os conjuntos de regras gerados pelos outros métodos. Ainda, foi verificado que em execuções específicas do método, as execuções que obtiveram os melhores desempenhos foram justamente aquelas nas quais foram gerados os menores conjuntos de regras. Adicionalmente, verificou-se que em algumas dessas execuções, os valores de $AU(\overline{PRC})$ obtidos foram superiores aos valores obtidos por todos os métodos investigados, inclusive o estado-da-arte. Isso mostra que o método proposto é capaz de gerar um conjunto de regras pequeno, e ainda assim obter desempenhos de classificação superiores aos desempenhos obtidos pelos métodos da literatura.

Considerando as curvas precisão-revocação de todos os métodos, pôde-se observar que as curvas dos métodos **HMC-GA** e **HMC-LMLP** foram bastante competitivas se comparadas com as curvas dos outros métodos investigados. Para as redes neurais, as curvas apresentadas são aquelas obtidas nas execuções que resultaram nos maiores valores de $AU(\overline{PRC})$ nos conjuntos de validação. Já para o algoritmo genético, as curvas são aquelas obtidas nas execuções que geraram os menores conjuntos de regras. É possível observar como o método **HMC-LMLP** obteve, em vários trechos das curvas, valores de precisão e revocação superiores aos valores obtidos pelos outros métodos, principalmente nas hierarquias estruturadas como árvores. Nas hierarquias estruturadas como grafos, observou-se um maior equilíbrio entre as curvas apresentadas, e pôde-se observar também como o método **HMC-LMLP** utilizando o algoritmo *Resilient Back-propagation* superou os outros métodos em alguns conjuntos de dados. As curvas apresentadas para o método **HMC-GA** também são bem competitivas se comparadas com as curvas obtidas pelos outros métodos da literatura, tanto nas hierarquias estruturadas como árvores quanto nas hierarquias estruturadas como grafos.

7.3 Trabalhos Futuros

Muitas pesquisas ainda podem ser realizadas como trabalhos futuros envolvendo problemas de classificação hierárquica multirrótulo, e também envolvendo os métodos de classificação propostos nesta Tese.

Considerando o método [HMC-LMLP](#), outros algoritmos podem ser utilizados no treinamento das redes neurais [MLP](#), como por exemplo o algoritmo *Improved Resilient Back-propagation* ([Riedmiller e Braun, 1993](#)), e o algoritmo Quickprop ([Fahlman, 1989](#)).

O procedimento de correção de inconsistências do método [HMC-LMLP](#) também pode ser aprimorado. Atualmente são realizados dois procedimentos um pouco diferentes para a garantia de consistência nas predições das hierarquias estruturadas como árvores e grafos. Em árvores, são simplesmente removidas das classes preditas aquelas classes cujas superclasses não foram preditas. Já em grafos, é verificado se algum caminho completo foi predito entre uma dada classe e o nó raiz. Se de todos os possíveis caminhos entre a classe e a raiz, pelo menos um deles tiver sido totalmente predito, então a classe, e todas as suas superclasses em todos os outros caminhos, são consideradas na classificação final do exemplo. Caso contrário, a classe em questão e todas as outras classes pertencentes aos caminhos inconsistentes entre a classe e a raiz são removidas da classificação final. Ao invés da realização desses procedimentos, pode-se, por exemplo, complementar os caminhos parcialmente preditos da hierarquia, ou então, utilizar um procedimento de votação por maioria para decidir quais classes serão preditas. Se por exemplo, de um caminho $A.B.C.D.E$, a classe B não for predita, e todas as outras classes forem preditas, o procedimento atual remove as classes C , D e E . Em uma votação por maioria, seria decidido pela complementação da classificação final com a classe B ao invés de remover todas as suas subclasses. Já se apenas as classes A e E fossem preditas, a votação por maioria resultaria na remoção da classe E .

A adaptação das hierarquias também pode sofrer aprimoramentos para permitir que todos os relacionamentos hierárquicos sejam utilizados durante o treinamento das redes neurais do método [HMC-LMLP](#). Uma das maneiras de se fazer isso seria replicar a mesma classe em diferentes níveis. Assim, dados dois caminhos $A.C$ e $A.B.C$, a classe C pertenceria aos níveis 2 e 3, e seria utilizada pelas redes [MLP](#) associadas ao segundo e terceiro níveis.

Considerando o número de camadas escondidas das redes neurais do método [HMC-LMLP](#), tem-se conhecimento das limitações práticas da utilização de apenas uma camada escondida, pois apesar do teorema da aproximação universal garantir que qualquer função contínua possa ser aprendida com apenas uma camada escondida, o teorema não garante que apenas uma camada seja suficiente para generalização ([Haykin, 1999](#)). Ainda, o teorema assume que a função contínua a ser aproximada é conhecida e que uma camada escondida com um número infinito de neurônios está disponível, o que na prática não ocorre.

Sabe-se que em redes neurais [MLP](#) com uma camada escondida, os neurônios tendem a interagir entre si de maneira global. Isso faz com que, em situações complexas, seja difícil melhorar a aproximação em um certo ponto sem a piorar em outro. Já com duas camadas escondidas, a

primeira extrai características locais, com alguns neurônios particionando o espaço de busca em regiões, e outros neurônios aprendendo características locais dessas regiões. A segunda camada então extrai características globais, pois seus neurônios combinam as saídas dos neurônios da camada anterior, operando em uma região particular do espaço (Haykin, 1999).

Mesmo tendo conhecimento das limitações da utilização de apenas uma camada escondida, optou-se pela utilização de apenas uma camada no método HMC-LMLP. Isso foi feito para verificar o desempenho das redes com apenas uma camada, e também para diminuir o tempo de treinamento. A utilização de duas camadas aumentaria muito o tempo de indução, principalmente nas hierarquias estruturadas como grafos. Como as redes com apenas uma camada escondida apresentaram resultados muito bons apesar das limitações práticas de sua utilização, pretende-se no futuro acrescentar mais uma camada escondida nas redes MLP do método, o que pode melhorar os resultados obtidos.

No método HMC-GA, uma das melhorias que devem ser feitas é o desenvolvimento de uma melhor função de *fitness*. Apesar de a função atual considerar boa uma regra que divide um conjunto de exemplos de maneira mais homogênea, ela apresenta uma falha quando o conjunto de exemplos é pequeno. No fim do procedimento de cobertura sequencial de exemplos, quando restam poucos exemplos de treinamento para serem cobertos, a função atual nunca permite que uma regra cubra todos os exemplos restantes. Isso acontece porque o conjunto de exemplos cobertos por tal regra possuiria valor de variância igual ao valor da variância de todos os exemplos, resultando em valor de *fitness* zero para a regra. Uma possível melhora seria incorporar ao *fitness* de uma regra o seu valor de $AU(\overline{PRC})$.

Outra melhora na função de *fitness* do algoritmo genético seria, de alguma maneira, verificar o quão bem as regras descobertas estão classificando os exemplos cobertos por elas. Isso poderia ser feito dividindo o conjunto de classes em grupos menores, e induzindo regras separadamente para cada conjunto. Isso tornaria o método não mais global, e sim local, aumentando o número de regras geradas. No entanto, controlando o número de grupos de classes, o número de regras geradas poderia também ser controlado para que a interpretação dos conjuntos de regras gerados não torne-se inviável. Esse procedimento talvez melhorasse o desempenho de classificação.

Ainda com o método HMC-GA, pretende-se estende-lo para permitir a geração de regras que comparem dois atributos diferentes entre si. Assim, serão geradas regras que contenham comparações do tipo $Atributo_i \geq Atributo_j$. Esse tipo de comparação aumenta muito o espaço de busca do método. No entanto, é preciso que se garanta que atributos do mesmo tipo estejam sendo comparados. Dessa forma, serão utilizados inicialmente os conjuntos de dados nos quais todos os atributos numéricos sejam dados de *microarray*, que são a maioria dos conjuntos de dados utilizados nesta pesquisa. Esses tipos de atributos podem ser comparados entre si depois de padronizados.

Por fim, outros domínios de aplicação relacionados à classificação hierárquica multirrótulo pretendem ser investigados, como a classificação de imagens (Barutcuoglu e DeCoro, 2006) e documentos (Rousu et al., 2006; Cesa-Bianchi et al., 2006; Mayne e Perry, 2009).

Referências Bibliográficas

- Aha, D. W., Kibler, D., e Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1):37–66. Citado nas páginas 37 e 40.
- Aleksovski, D., Kocev, D., e Dzeroski, S. (2009). Evaluation of distance measures for hierarchical multilabel classification in functional genomics. In *Workshop on Learning from Multi-Label Data*, páginas 5–16. Citado nas páginas 43 e 48.
- Alves, R., Delgado, M., e Freitas, A. (2008). Multi-label Hierarchical Classification of Protein Functions with Artificial Immune Systems. In *Brazilian Symposium on Bioinformatics*, volume 5167 of *Lecture Notes in Bioinformatics*, páginas 1–12, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas 45, 46, e 48.
- Alves, R., Delgado, M., e Freitas, A. (2010). Knowledge discovery with Artificial Immune Systems for hierarchical multi-label classification of protein functions. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, volume 1, páginas 2097–2104, Barcelona. IEEE. Citado nas páginas 31, 46, e 48.
- Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., e Sherlock, G. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature Genetics*, 25(1):25–29. Citado nas páginas 3, 37, e 96.
- Bäck, T., Fogel, D., e Michalewicz, Z. (2000a). *Evolutionary Computation 1: Basic Algorithms and Operators*. Institute of Physics Publishing, Bristol. Citado na página 10.
- Bäck, T., Fogel, D., e Michalewicz, Z. (2000b). *Evolutionary Computation 2: Advanced Algorithms and Operators*. Institute of Physics Publishing, Bristol. Citado na página 10.
- Bairoch, A. e Apweiler, R. (2000). The SWISS-PROT protein sequence database and its supplement TrEMBL. *Nucleic Acids Research*, 28(1):45–48. Citado na página 58.

- Baraniuk, R. G. e Jones, D. L. (1994). A Signal-Dependent Time-Frequency Representation: Fast Algorithm for Optimal Kernel Design. *IEEE Transactions on Signal Processing*, 42(1):134–146. Citado na página 41.
- Barutcuoglu, Z. e DeCoro, C. (2006). Hierarchical Shape Classification Using Bayesian Aggregation. In *International Conference on Shape Modeling and Applications*, página 44, Los Alamitos, CA, USA. IEEE Computer Society. Citado nas páginas 37, 48, e 136.
- Barutcuoglu, Z., Schapire, R. E., e Troyanskaya, O. G. (2006). Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836. Citado nas páginas 37 e 48.
- Bayes, T. (1763). An essay towards solving a problem in doctrine of chances. *Philos. Trans. R. Soc. London*, 53:293–315. Citado na página 37.
- Bennett, P. N. e Nguyen, N. (2009). Refined experts: improving classification in large taxonomies. In *International Conference on Research and Development in Information Retrieval*, páginas 11–18, New York, NY, USA. ACM. Citado nas páginas 38 e 48.
- Beyer, H.-G. e Schwefel, H.-P. (2002). Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1):3–52. Citado na página 10.
- Bi, W. e Kwok, J. (2011). Multi-Label Classification on Tree- and DAG-Structured Hierarchies. In Getoor, L. e Scheffer, T., editors, *International Conference on Machine Learning, ICML'11*, páginas 17–24, New York, NY, USA. ACM. Citado nas páginas 41 e 48.
- Bi, W. e Kwok, J. T. (2012). Mandatory leaf node prediction in hierarchical multilabel classification. In *Advances in Neural Information Processing Systems*, páginas 153–161. Citado nas páginas 45 e 48.
- Blanchard, E., Harzallah, M., Bri, H., Kuntz, P., e Pauc, R. C. (2005). A Typology Of Ontology-Based Semantic Measures. In *Proceedings of EMOI-INTEROP workshop*, páginas 13–14. Citado na página 53.
- Blockeel, H., Bruynooghe, M., Dzeroski, S., Ramon, J., e Struyf, J. (2002). Hierarchical multi-classification. In *Workshop on Multi-Relational Data Mining*, páginas 21–35. Citado nas páginas 42 e 48.
- Blockeel, H., De Raedt, L., e Ramon, J. (1998). Top-down induction of clustering trees. In *International Conference on Machine Learning*, páginas 55–63. Morgan Kaufmann. Citado nas páginas 42 e 95.
- Blockeel, H., Schietgat, L., Struyf, J., Dzeroski, S., e Clare, A. (2006). Decision Trees for Hierarchical Multilabel Classification: A Case Study in Functional Genomics. In Fürnkranz, J., Scheffer, T., e Spiliopoulou, M., editors, *PKDD*, volume 4213 of *LNCS*, páginas 18–29. Springer. Citado nas páginas 42 e 48.

- Booker, L. B., Fogel, D. B., Whitley, D., , e Angeline, P. (1997). Recombination. In Bäck, T., Fogel, D. B., e Michalewicz, Z., editors, *The Handbook of Evolutionary Computation*, chapter E3.3, páginas C3.3:1–C3.3:27. IOP Publishing and Oxford University Press, Philadelphia, PA. Citado na página 14.
- Borges, H. e Nievola, J. (2012). Multi-label hierarchical classification using a competitive neural network for protein function prediction. In *International Joint Conference on Neural Networks*, páginas 1–8. Citado nas páginas 47 e 48.
- Boutell, M. R., Luo, J., Shen, X., e Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771. Citado nas páginas 33, 40, e 45.
- Breiman, L. (1996). Bagging Predictors. *Machine Learning*, 24(2):123–140. Citado na página 44.
- Breiman, L. (2001). Random Forests. *Machine Learning*, 45:5–32. Citado na página 44.
- Brucker, F., Benites, F., e Sapozhnikova, E. (2010). Multi-label classification and extracting predicted class hierarchies. *Pattern Recognition*, 44:724–738. Citado na página 46.
- Cai, L. e Hofmann, T. (2004). Hierarchical document categorization with support vector machines. In *International conference on Information and knowledge management*, CIKM '04, páginas 78–87, New York, NY, USA. ACM. Citado nas páginas 44 e 48.
- Cai, L. e Hofmann, T. (2007). Exploiting known taxonomies in learning overlapping concepts. In *International joint conference on Artificial intelligence*, páginas 714–719, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Citado nas páginas 43, 45, e 48.
- Carpenter, G. A. e Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115. Citado na página 47.
- Carvalho, A. e Freitas, A. (2009). *A tutorial on multi-label classification techniques*, volume 5 of *Studies in Computational Intelligence* 205, páginas 177–195. Springer. Citado na página 32.
- Carvalho, R., Brunoro, G., e Pappa, G. (2011). HCGA: A genetic algorithm for hierarchical classification. In *IEEE Congress on Evolutionary Computation*, páginas 933–940. Citado na página 116.
- Cerri, R. e Carvalho, A. C. P. L. F. (2010a). Hierarchical Multilabel Classification Using Top-Down Label Combination and Artificial Neural Networks. In *Proceeding of the XI Brazilian Symposium on Artificial Neural Network*, páginas 253–258. IEEE. Citado nas páginas 45 e 48.

- Cerri, R. e Carvalho, A. C. P. L. F. (2010b). New top-down methods using SVMs for Hierarchical Multilabel Classification Problems. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, páginas 3064–3071, Barcelona. IEEE. Citado nas páginas 40, 45, e 48.
- Cerri, R., de Carvalho, A., e Freitas, A. (2011). Adapting non-hierarchical multilabel classification methods for hierarchical multilabel classification. *Intelligent Data Analysis*, 15(6):861–887. Citado nas páginas 40, 45, e 48.
- Cerri, R., Pappa, G. L., Carvalho, A. C. P., e Freitas, A. A. (2013). An extensive evaluation of decision tree-based hierarchical multilabel classification methods and performance measures. *Computational Intelligence*. Aceito para publicação. Citado nas páginas 61, 62, e 95.
- Cesa-Bianchi, N., Gentile, C., e Zaniboni, L. (2006). Incremental Algorithms for Hierarchical Classification. *Machine Learning*, 7:31–54. Citado nas páginas 1, 37, 38, 39, 48, 50, 51, 52, e 136.
- Cesa-Bianchi, N., Re, M., e Valentini, G. (2011). Synergy of multi-label hierarchical ensembles, data fusion, and cost-sensitive methods for gene functional inference. *Machine Learning*, páginas 1–33. Citado nas páginas 41, 42, e 48.
- Cesa-Bianchi, N. e Valentini, G. (2010). Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. *Journal of Machine Learning Research*, 8:14–29. Citado nas páginas 38, 40, 42, e 48.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., e Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357. Citado na página 39.
- Chen, B. e Hu, J. (2010). Hierarchical Multi-label Classification incorporating prior information for gene function prediction. In *International Conference on Intelligent Systems Design and Applications*, páginas 231–236. IEEE. Citado nas páginas 39 e 48.
- Chen, Y., Li, Z., Hu, X., e Liu, J. (2010). Hierarchical classification with dynamic-threshold SVM ensemble for gene function prediction. In *Proceedings of the International Conference on Advanced data mining and applications*, ADMA'10, páginas 336–347, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas 41 e 48.
- Cherman, E. A., Metz, J., e Monard, M. C. (2012). Incorporating label dependency into the binary relevance framework for multi-label classification. *Expert Systems with Applications*, 39(2):1647–1655. Citado na página 68.
- Clare, A. e King, R. D. (2003). Predicting gene function in *Saccharomyces cerevisiae*. *Bioinformatics*, 19:42–49. Citado nas páginas 41, 43, e 48.

- Cohen, W. W. (1995). Fast effective rule induction. In *Twelfth International Conference on Machine Learning*, páginas 115–123. Citado na página 40.
- Crammer, K. e Singer, Y. (2002). On the algorithmic implementation of multiclass kernel-based vector machines. *Machine Learning*, 2:265–292. Citado na página 44.
- Darwin, C. ([1872?]). *The origin of species by means of natural selection*. New York: Hurst,. Citado na página 9.
- Davis, J. e Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. In *International conference on Machine learning*, páginas 233–240, New York, NY, USA. ACM. Citado nas páginas 53 e 94.
- de Castro, L. N. (2006). *Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications*. Chapman & Hall/CRC. Citado nas páginas 9, 10, 11, 13, 15, 16, 17, e 21.
- de Castro, L. N. e Timmis, J. I. (2002). *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer-Verlag, London. Citado na página 45.
- DeCoro, C., Barutcuoglu, Z., e Fiebrink, R. (2007). Bayesian Aggregation for Hierarchical Genre Classification. In *International Symposium on Music Information Retrieval 2007*. Citado nas páginas 37 e 48.
- Dekel, O., Keshet, J., e Singer, Y. (2004). Large margin hierarchical classification. In *International Conference on Machine Learning, ICML '04*, páginas 27–, New York, NY, USA. ACM. Citado na página 45.
- Dembczynski, K., Cheng, W., e Hüllermeier, E. (2010). Bayes optimal multilabel classification via probabilistic classifier chains. In Fürnkranz, J. e Joachims, T., editors, *Proceedings of the 27th International Conference on Machine Learning*, páginas 279–286. Omnipress. Citado na página 68.
- Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7:1–30. Citado na página 94.
- Díez, J., del Coz, J. J., e Bahamonde, A. (2010). A semi-dependent decomposition approach to learn hierarchical classifiers. *Pattern Recognition*, 43:3795–3804. Citado nas páginas 40 e 48.
- Dimitrovski, I., Kocev, D., Loskovska, S., e Deroski, S. (2011). Hierarchical annotation of medical images. *Pattern Recognition*, 44:2436–2449. Citado nas páginas 44 e 48.
- Dimitrovski, I., Kocev, D., Loskovska, S., e Džeroski, S. (2009). ImageCLEF 2009 medical image annotation task: PCTs for hierarchical multi-label classification. In *International conference on Cross-language evaluation forum: multimedia experiments, CLEF'09*, páginas 231–238, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas 44 e 48.

- Dimitrovski, I., Kocev, D., Loskovska, S., e Džeroski, S. (2010). Detection of visual concepts and annotation of images using ensembles of trees for hierarchical multi-label classification. In *International Conference on Recognizing Patterns in Signals, Speech, Images, and Videos*, ICPR'10, páginas 152–161, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas 44 e 48.
- Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. Tese de Doutorado, Dipartimento di Elettronica, Politecnico di Milano, IT. Citado na página 46.
- Dorigo, M., Maniezzo, V., e Colorni, A. (1991). Positive feedback as a search strategy. Technical report, Dipartimento di Elettronica, Politecnico di Milano, IT. Citado na página 46.
- Dumais, S. e Chen, H. (2000). Hierarchical classification of Web content. In *International Conference on Research and Development in Information Retrieval*, SIGIR '00, páginas 256–263, New York, NY, USA. ACM. Citado nas páginas 36 e 48.
- Eiben, A. E. e Smith, J. E. (2003). *Introduction to Evolutionary Computing*. Springer Verlag. Citado na página 11.
- Eisner, R., Poulin, B., Szafron, D., Lu, P., e Greiner, R. (2005). Improving Protein Function Prediction using the Hierarchical Structure of the Gene Ontology. In *Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, páginas 1–10. Citado na página 49.
- Elisseeff, A. e Weston, J. (2001). Kernel Methods for Multi-labelled Classification and Categorical Regression Problems. In *Advances in Neural Information Processing Systems*, páginas 681–687. MIT Press. Citado na página 44.
- Esuli, A., Fagni, T., e Sebastiani, F. (2006). TreeBoost.MH: A boosting algorithm for multi-label hierarchical text categorization. In *International Symposium on String Processing and Information Retrieval*, páginas 13–24. Citado nas páginas 40 e 48.
- Esuli, A., Fagni, T., e Sebastiani, F. (2008). Boosting multi-label hierarchical text categorization. *Information Retrieval*, 11(4):287–313. Citado nas páginas 40 e 48.
- Fahlman, S. E. (1989). Faster-learning variations on back-propagation: An empirical study. In Touretzky, D. S., Hinton, G. E., e Sejnowski, T. J., editors, *Proceedings of the 1988 Connectionist Models Summer School*, páginas 38–51. San Francisco, CA: Morgan Kaufmann. Citado na página 135.
- Fogel, D. B. (2000). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. The IEEE Press, 2 edition. Citado na página 12.
- Freitas, A. A. e Carvalho, A. C. (2007). *A Tutorial on Hierarchical Classification with Applications in Bioinformatics*, volume 1, chapter VII, páginas 175–208. Idea Group. Research and Trends in Data Mining Technologies and Applications. Citado na página 30.

- Friedman, N., Geiger, D., e Goldszmidt, M. (1997). Bayesian Network Classifiers. *Mach. Learn.*, 29(2-3):131–163. Citado nas páginas [37](#) e [40](#).
- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition. Citado nas páginas [12](#) e [14](#).
- Goldberg, D. E. e Holland, J. H. (1988). Genetic Algorithms and Machine Learning. *Machine Learning*, 3:95–99. Citado na página [9](#).
- Guan, Y., Myers, C., Hess, D., Barutcuoglu, Z., Caudy, A., e Troyanskaya, O. (2008). Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biology*, 9(Suppl 1):S3. Citado nas páginas [37](#) e [48](#).
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., e Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18. Citado na página [104](#).
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109. Citado na página [38](#).
- Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition. Citado nas páginas [15](#), [18](#), [19](#), [20](#), [21](#), [22](#), [24](#), [26](#), [135](#), e [136](#).
- He, J. e Yao, X. (2003). Towards an analytic framework for analysing the computation time of evolutionary algorithms. *Artificial Intelligence*, 145:59 – 97. Citado na página [89](#).
- Hecht-Nielsen, R. (1989). *Neurocomputing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. Citado na página [15](#).
- Hersh, W., Buckley, C., Leone, T. J., e Hickam, D. (1994). OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *International Conference on Research and Development in Information Retrieval*, páginas 192–201, New York, NY, USA. Springer-Verlag New York, Inc. Citado na página [37](#).
- Hoerl, A. E. e Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67. Citado na página [41](#).
- Holden, N. e Freitas, A. (2006). Hierarchical Classification of G-Protein-Coupled Receptors with a PSO/ACO Algorithm. In *Swarm Intelligence Symposium*, páginas 77–84. IEEE Press. Citado na página [59](#).
- Holland, J. H. (1975). *Adaptation in natural and artificial systems*. MIT Press. Citado nas páginas [10](#), [11](#), e [12](#).

- Igel, C. e Husken, M. (2000). Improving the rprop learning algorithm. In *International Symposium on Neural Computation*. Citado na página [28](#).
- Ipeirotis, P. G., Gravano, L., e Sahami, M. (2001). Probe, Count, and Classify: Categorizing Hidden Web Databases. In *SIGMOD Conference*, páginas 67–78. Citado na página [52](#).
- Itskevitch, J. (2001). Automatic Hierarchical E-Mail Classification Using Association Rules. Dissertação de Mestrado, Simon Fraser University. Citado na página [56](#).
- Jiang, J. e Conrath, D. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *International Conference on Research in Computational Linguistics*, páginas 19–33. Citado na página [58](#).
- Jin, B., Muller, B., Zhai, C., e Lu, X. (2008). Multi-label literature classification based on the Gene Ontology graph. *BMC Bioinformatics*, 9(1):525. Citado nas páginas [38](#), [48](#), [49](#), e [58](#).
- Kiritchenko, S., Matwin, S., e Famili, A. (2005). Functional annotation of genes using hierarchical text categorization. In *Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*. Citado nas páginas [43](#), [48](#), [49](#), [52](#), [54](#), [56](#), [62](#), e [94](#).
- Kiritchenko, S., Matwin, S., e Famili, A. F. (2004). Hierarchical Text Categorization as a Tool of Associating Genes with Gene Ontology Codes. In *European Workshop on Data Mining and Text Mining in Bioinformatics*, páginas 30–34, Pisa, Italy. Citado nas páginas [40](#), [43](#), [48](#), [49](#), e [50](#).
- Kiritchenko, S., Matwin, S., Nock, R., e Famili, A. (2006). Learning and Evaluation in the Presence of Class Hierarchies: Application to Text Categorization. In Lamontagne, L. e Marchand, M., editors, *Advances in Artificial Intelligence*, volume 4013 of *Lecture Notes in Computer Science*, páginas 395–406. Springer Berlin Heidelberg. Citado nas páginas [43](#) e [48](#).
- Kirkpatrick, S., Gelatt, C. D., e Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 220(4598):671–680. Citado na página [38](#).
- Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press. Citado na página [10](#).
- Krauth, W. e Mezard, M. (1987). Learning algorithms with optimal stability in neural networks. *Journal of Physics A: Mathematical and General*, 20(11):L745. Citado na página [45](#).
- Langley, P., Iba, and, W., e Thompson, K. (1992). An analysis of Bayesian classifiers. In *National Conference on Artificial intelligence*, AAAI'92, páginas 223–228. AAAI Press. Citado na página [37](#).

- Lewis, D. D., Yang, Y., Rose, T. G., e Li, F. (2004). RCV1: A New Benchmark Collection for Text Categorization Research. *Journal Machine Learning Research*, 5:361–397. Citado nas páginas 37 e 41.
- Lin, D. (1998). An Information-Theoretic Definition of Similarity. In *International Conference on Machine Learning*, ICML '98, páginas 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Citado na página 58.
- Lord, P., Stevens, R., Brass, A., e Goble, C. (2003). Investigating semantic similarity measures across the gene ontology: The relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283. Citado nas páginas 58 e 59.
- Manning, C. D. e Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA. Citado na página 62.
- Mayne, A. e Perry, R. (2009). Hierarchically classifying documents with multiple labels. In *Symposium on Computational Intelligence and Data Mining*, páginas 133–139. Citado nas páginas 1, 38, 48, e 136.
- McClelland, J. L., Rumelhart, D. E., e the PDP Research Group (1986). *Parallel distributed processing: explorations in the microstructure of cognition, vol. 2: psychological and biological models*. MIT Press, Cambridge, MA, USA. Citado na página 21.
- McCulloch, W. e Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, 5:115–133. 10.1007/BF02478259. Citado na página 18.
- MedLine database (2008). Medline database. URL: <http://www.ncbi.nlm.nih.gov/pubmed/>. Citado na página 40.
- Mewes, H. W., Frishman, D., Güldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgens-tern, B., Münsterkötter, M., Rudd, S., e Weil, B. (2002). MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 30(1):31–34. Citado na página 39.
- Minsky, M. e Papert, S. (1969). *Perceptrons*. MIT Press. Citado nas páginas 20 e 21.
- Nguyen, C., Dung, T., e Cao, T. (2005). Text classification for DAG-structured categories. *Lecture Notes in Computer Science*, 3518:290–300. Citado nas páginas 36 e 48.
- Nigrin, A. (1993). *Neural networks for pattern recognition*. MIT Press. Citado na página 15.
- Otero, F., Freitas, A., e Johnson, C. (2009). A hierarchical classification ant colony algorithm for predicting gene ontology terms. In Pizzuti, C., Ritchie, M., e Giacobini, M., editors, *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, volume Lecture Notes in Computer Science 5483, páginas 68–79. Springer. Citado na página 46.

- Otero, F. E. B., Freitas, A. A., e Johnson, C. G. (2010). A hierarchical multi-label classification ant colony algorithm for protein function prediction. *Memetic Computing*, 2(3):165–181. Citado nas páginas [3](#), [31](#), [46](#), [48](#), [62](#), e [89](#).
- Platt, J. C. (1999). Probabilistic Outputs for Support Vector Machines and Comparison to Regularized Likelihood Methods. *Advances in Kernel Methods Support Vector Learning*, páginas 61–74. Citado na página [39](#).
- Pugelj, M. e Džeroski, S. (2011). Predicting structured outputs k-nearest neighbours method. In *Proceedings of the 14th international conference on Discovery science*, páginas 262–276, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas [44](#) e [48](#).
- Punera, K. e Rajan, S. (2009). Improved Multi Label Classification in Hierarchical Taxonomies. In *International Conference on Data Mining Workshops, ICDMW '09*, páginas 388–393, Washington, DC, USA. IEEE Computer Society. Citado nas páginas [38](#) e [48](#).
- Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. Citado na página [40](#).
- Raghavan, V., Bollmann, P., e Jung, G. S. (1989). A critical investigation of recall and precision as measures of retrieval system performance. *ACM Trans. Inf. Syst.*, 7:205–229. Citado na página [62](#).
- Re, M. e Valentini, G. (2010a). An Experimental Comparison of Hierarchical Bayes and True Path Rule Ensembles for Protein Function Prediction. In Gayar, N. E., Kittler, J., e Roli, F., editors, *International Workshop on Multiple Classifier Systems*, volume 5997 of *Lecture Notes in Computer Science*, páginas 294–303. Springer. Citado nas páginas [40](#) e [48](#).
- Re, M. e Valentini, G. (2010b). Simple ensemble methods are competitive with state-of-the-art data integration methods for gene function prediction. *Journal of Machine Learning Research - Proceedings Track*, 8:98–111. Citado na página [41](#).
- Read, J., Pfahringer, B., Holmes, G., e Frank, E. (2009). Classifier chains for multi-label classification. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II, ECML PKDD '09*, páginas 254–269, Berlin, Heidelberg. Springer-Verlag. Citado na página [68](#).
- Riedmiller, M. e Braun, H. (1993). A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In *International Conference on Neural Networks*, volume 1, páginas 586–591. Citado nas páginas [26](#), [27](#), [28](#), [101](#), e [135](#).
- Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408. Citado na página [20](#).
- Rosenblatt, F. (1962). *Principles of Neurodynamics*. Spartan Book. Citado na página [20](#).

- Rousu, J., Saunders, C., Szedmak, S., e Shawe-Taylor, J. (2006). Kernel-Based Learning of Hierarchical Multilabel Classification Models. *Journal of Machine Learning Research*, 7:1601–1626. Citado nas páginas 43, 48, e 136.
- Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Güldener, U., Mannhaupt, G., Münsterkötter, M., e Mewes, H. W. (2004). The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Research*, 32(18):5539–5545. Citado nas páginas 3, 39, e 96.
- Rumelhart, D. E. e McClelland, J. L. (1986). *Parallel distributed processing: explorations in the microstructure of cognition*, volume 1: foundations. MIT Press, Cambridge, MA, USA. Citado nas páginas 18, 21, 24, e 101.
- Sangsuriyun, S., Marukatat, S., e Waiyamai, K. (2010). Hierarchical Multi-label Associative Classification (HMAC) using negative rules. In *International Conference on Cognitive Informatics*, páginas 919–924. IEEE. Citado nas páginas 44, 48, e 52.
- Sapozhnikova, E. P. (2009). ART-Based Neural Networks for Multi-label Classification. In *International Symposium on Intelligent Data Analysis, IDA '09*, páginas 167–177, Berlin, Heidelberg. Springer-Verlag. Citado na página 47.
- Sastry, K., Goldberg, D., e Kendall, G. (2005). Genetic Algorithms. In Burke, E. K. e Kendall, G., editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Methodologies*, chapter 4, páginas 97–125. Springer. Citado nas páginas 12, 14, e 15.
- Schapire, R. E. e Singer, Y. (1999). Improved Boosting Algorithms Using Confidence-rated Predictions. In *Machine Learning*, volume 37, páginas 297–336, Hingham, MA, USA. Kluwer Academic Publishers. Citado na página 40.
- Schapire, R. E. e Singer, Y. (2000). BoosTexter: a boosting-based system for text categorization. In *Machine Learning*, volume 39, páginas 135–168, Hingham, MA, USA. Kluwer Academic Publishers. Citado na página 47.
- Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., e Dzeroski, S. (2010). Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics*, 11(2). Citado nas páginas 44 e 48.
- Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47. Citado na página 50.
- Shen, X., Boutell, M., Luo, J., e Brown, C. (2003). Multilabel machine learning and its application to semantic scene classification. In Yeung, M. M., Lienhart, R. W., e Li, C.-S., editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 5307 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, páginas 188–199. Citado nas páginas 32 e 40.

- Silla, C. e Freitas, A. (2010). A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22:31–72. Citado nas páginas [1](#), [2](#), [30](#), [35](#), [47](#), [48](#), [62](#), [71](#), e [95](#).
- Spears, W. (1997). Recombination parameters. In Bäck, T., Fogel, D. B., e Michalewicz, Z., editors, *The Handbook of Evolutionary Computation*, chapter E1.3, páginas E1.3:1–E1.3:13. IOP Publishing and Oxford University Press, Philadelphia, PA. Citado na página [14](#).
- Struyf, J., Blockeel, H., e Clare, A. (2005). Hierarchical multi-classification with predictive clustering trees in functional genomics. In *Workshop on Computational Methods in Bioinformatics*, volume 3808 of *LNAI*, páginas 272–283. Springer Berlin / Heidelberg. Citado nas páginas [42](#) e [48](#).
- Sun, A. e Lim, E.-P. (2001). Hierarchical Text Classification and Evaluation. In *International Conference on Data Mining*, páginas 521–528. Citado nas páginas [30](#), [31](#), [36](#), [48](#), [54](#), [55](#), e [57](#).
- Sun, A., Lim, E.-P., e Ng, W.-K. (2003). Performance measurement framework for hierarchical text classification. *Journal of the American Society for Information Science and Technology*, 54(11):1014–1028. Citado nas páginas [30](#), [36](#), [48](#), [54](#), e [57](#).
- Sun, A., peng Lim, E., Ng, W. K., e Srivastava, J. (2004). Blocking Reduction Strategies in Hierarchical Text Classification. *IEEE Transactions on Knowledge and Data Engineering*, 16:1305–1308. Citado nas páginas [30](#), [36](#), [37](#), [41](#), e [48](#).
- Tan, A.-H. (1995). Adaptive resonance associative map. *Neural Netw.*, 8:437–446. Citado na página [47](#).
- Taskar, B., Guestrin, C., e Koller, D. (2003). Max-Margin Markov Networks. In Thrun, S., Saul, L. K., e Schölkopf, B., editors, *Advances in Neural Information Processing Systems*, páginas 25–32. MIT Press. Citado na página [43](#).
- Toutanova, K., Chen, F., Papat, K., e Hofmann, T. (2001). Text classification in a hierarchical mixture model for small training sets. In *International Conference on Information and Knowledge Management*, páginas 105–113, New York, NY, USA. ACM. Citado na página [40](#).
- Trappenberg, T. P. (2002). *Fundamentals of Computational Neuroscience*. Oxford U. Press. Citado nas páginas [16](#) e [17](#).
- Tsochantaridis, I., Joachims, T., Hofmann, T., e Altun, Y. (2005). Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research*, 6:1453–1484. Citado nas páginas [41](#) e [43](#).
- Tsoumakas, G., Katakis, I., e Vlahavas, I. P. (2010). Mining Multi-label Data. In Maimon, O. e Rokach, L., editors, *Data Mining and Knowledge Discovery Handbook*, páginas 667–685. Springer, 2nd edition. Citado na página [33](#).

- Tsoumakas, G. e Vlahavas, I. (2007). Random k-Labelsets: An Ensemble Method for Multilabel Classification. In *European Conference on Machine Learning*, páginas 406–417, Warsaw, Poland. Citado nas páginas [40](#) e [45](#).
- Valentini, G. (2009). True Path Rule Hierarchical Ensembles. In *International Workshop on Multiple Classifier Systems*, volume 5519 of *Lecture Notes in Bioinformatics*, páginas 232–241, Berlin, Heidelberg. Springer-Verlag. Citado nas páginas [39](#) e [48](#).
- Valentini, G. (2011). True Path Rule Hierarchical Ensembles for Genome-Wide Gene Function Prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:832–847. Citado nas páginas [1](#), [39](#), [42](#), e [48](#).
- Valentini, G. e Re, M. (2009). Weighted true path rule: a multilabel hierarchical algorithm for gene function prediction. In *Workshop on Learning from Multi-Label Data*, páginas 132–145. Citado nas páginas [39](#), [40](#), e [48](#).
- Vapnik, V. N. (1999). *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer-Verlag New York, Inc. Citado nas páginas [30](#) e [36](#).
- Vens, C., Struyf, J., Schietgat, L., Džeroski, S., e Blockeel, H. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214. Citado nas páginas [1](#), [3](#), [31](#), [34](#), [41](#), [42](#), [43](#), [44](#), [48](#), [53](#), [54](#), [62](#), [81](#), [89](#), [95](#), e [100](#).
- Verspoor, K., Cohn, J., Mniszewski, S., e Joslyn, C. (2006). A categorization approach to automated ontological function annotation. *Protein Science*, 15(6):1544–1549. Citado na página [49](#).
- Wang, K. e Senqiang, K. W. (2001). Hierarchical Classification of Real Life Documents. In *International Conference on Data Mining*. Citado na página [59](#).
- Wang, X., Zhao, H., e liang Lu, B. (2011). Enhance k-nearest neighbour algorithm for large-scale multi-labeled hierarchical classification. In *Second Pascal Challenge on Large Scale Hierarchical Text classification*. Citado nas páginas [44](#) e [48](#).
- Webb, E. C., of Biochemistry, I. U., e Biology., M. (1992). *Enzyme nomenclature 1992 : recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes / prepared for NC-IUBMB by Edwin C. Webb*. Published for the International Union of Biochemistry and Molecular Biology by Academic Press, San Diego. Citado na página [43](#).
- Wetzel, A. (1983). *Evaluation of the Effectiveness of Genetic Algorithms in Combinatorial Optimization*. Tese de Doutorado, University of Pittsburgh, Pittsburgh, PA. Citado nas páginas [12](#) e [13](#).
- WIPO (2001). World Intellectual Property Organization. <http://www.wipo.int/portal/index.html.en>. Citado nas páginas [41](#) e [43](#).

- Woolam, C. e Khan, L. (2008). Multi-label large margin hierarchical perceptron. *International Journal of Data Mining, Modelling and Management*, 1(1):5–22. Citado nas páginas 45 e 48.
- Yang, Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval*, 1(1/2):69–90. Citado na página 54.
- Zadrozny, B. e Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *International Conference on Knowledge Discovery and Data Mining*, KDD '02, páginas 694–699, New York, NY, USA. ACM. Citado na página 39.
- Zhang, M.-L. e Zhou, Z.-H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40:2038–2048. Citado na página 47.

Resultados por Nível das Variações do método HMC-LMLP

Este Apêndice apresenta uma comparação de todas as variações do método [HMC-LMLP](#) quanto aos desempenhos obtidos em cada nível. As Tabelas [A.1](#), [A.2](#), [A.3](#), [A.4](#), [A.5](#) e [A.6](#) apresentam as médias das áreas abaixo das curvas [PR](#) individuais ($AUPRC$) obtidas em cada classe, por nível hierárquico. Não são mostrados os resultados para o primeiro nível porque as variações do método [HMC-LMLP](#) são diferentes apenas do segundo nível em diante. É importante ter em mente que a avaliação feita nível a nível é diferente das avaliações utilizando as medidas $AU(\overline{PRC})$ e $\overline{AUPRC}_w$. Isso significa que não foi calculada nenhuma dessas medidas em cada nível, mas sim foi feita, em cada nível, a média das $AUPRC$ individuais obtidas em cada classe.

Como pode ser observado nas Tabelas [A.1](#) e [A.2](#), as variações que utilizam as predições (Bp-Predicted e Rp-Predicted) obtiveram os melhores resultados na maioria dos níveis hierárquicos da maioria dos conjuntos de dados estruturados como árvores. Apesar dos valores obtidos serem muito baixos em cada nível, eles mostram que a utilização das predições para complementar os vetores de atributos dos exemplos melhorou a classificação nos níveis mais profundos das hierarquias. Os baixos valores obtidos também mostram o quão difícil é a tarefa de classificação nos níveis mais profundos.

Considerando os conjuntos de dados estruturados como grafos (Tabelas [A.3](#), [A.4](#), [A.5](#) e [A.6](#) do Apêndice [A](#)), observa-se um maior equilíbrio nos resultados obtidos pelas diferentes variações do método [HMC-LMLP](#). Quando são comparadas as variações utilizando o algoritmo *Back-propagation*, observa-se que a variação Bp-Predicted obteve valores ligeiramente melhores na maioria dos níveis na maioria dos conjuntos de dados. Já considerando as variações utilizando o algoritmo *Resilient Back-propagation*, observa-se um maior equilíbrio entre os desempenhos obtidos pelas variações Rp-Predicted, Rp-True e Rp-NoLabels.

Tabela A.1: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados estruturados como árvores, utilizando o algoritmo *Back-propagation*

Métodos		Conjunto de dados Cellcycle			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.043 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.081 ± 0.004	0.030 ± 0.001	0.017 ± 0.001	0.004 ± 0.001	0.011 ± 0.004
Bp-True	0.072 ± 0.001	0.026 ± 0.001	0.014 ± 0.001	0.004 ± 0.002	0.006 ± 0.003
Bp-NoLabels	0.080 ± 0.003	0.027 ± 0.001	0.014 ± 0.001	0.004 ± 0.001	0.007 ± 0.003
Métodos		Conjunto de dados Church			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.040 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.051 ± 0.001	0.018 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Bp-True	0.050 ± 0.001	0.017 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-NoLabels	0.051 ± 0.001	0.016 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Métodos		Conjunto de dados Derisi			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.042 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.063 ± 0.002	0.022 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.002 ± 0.001
Bp-True	0.058 ± 0.001	0.018 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-NoLabels	0.063 ± 0.001	0.019 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Métodos		Conjunto de dados Eisen			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.049 ± 0.001	0.018 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.113 ± 0.005	0.043 ± 0.003	0.025 ± 0.002	0.006 ± 0.001	0.001 ± 0.001
Bp-True	0.091 ± 0.003	0.034 ± 0.003	0.020 ± 0.002	0.005 ± 0.001	0.006 ± 0.017
Bp-NoLabels	0.107 ± 0.005	0.035 ± 0.001	0.018 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Métodos		Conjunto de dados Gasch1			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.045 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.115 ± 0.003	0.038 ± 0.002	0.019 ± 0.001	0.008 ± 0.001	0.172 ± 0.026
Bp-True	0.103 ± 0.005	0.033 ± 0.002	0.017 ± 0.002	0.006 ± 0.001	0.101 ± 0.041
Bp-NoLabels	0.114 ± 0.002	0.036 ± 0.003	0.018 ± 0.003	0.009 ± 0.004	0.108 ± 0.047
Métodos		Conjunto de dados Gasch2			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.043 ± 0.001	0.015 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.093 ± 0.002	0.031 ± 0.001	0.016 ± 0.001	0.006 ± 0.001	0.083 ± 0.037
Bp-True	0.073 ± 0.003	0.025 ± 0.002	0.013 ± 0.001	0.004 ± 0.001	0.024 ± 0.038
Bp-NoLabels	0.087 ± 0.006	0.026 ± 0.001	0.012 ± 0.001	0.004 ± 0.001	0.011 ± 0.005
Métodos		Conjunto de dados Pheno			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.041 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.052 ± 0.002	0.016 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Bp-True	0.045 ± 0.002	0.015 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Bp-NoLabels	0.051 ± 0.001	0.016 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Métodos		Conjunto de dados Spo			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.042 ± 0.001	0.015 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.062 ± 0.001	0.023 ± 0.001	0.012 ± 0.001	0.004 ± 0.001	0.009 ± 0.004
Bp-True	0.059 ± 0.002	0.021 ± 0.001	0.011 ± 0.001	0.003 ± 0.001	0.003 ± 0.001
Bp-NoLabels	0.062 ± 0.002	0.021 ± 0.001	0.010 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Métodos		Conjunto de dados Expr			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.044 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.128 ± 0.004	0.047 ± 0.003	0.025 ± 0.001	0.010 ± 0.001	0.203 ± 0.023
Bp-True	0.123 ± 0.007	0.045 ± 0.006	0.024 ± 0.003	0.010 ± 0.002	0.190 ± 0.058
Bp-NoLabels	0.125 ± 0.007	0.044 ± 0.004	0.024 ± 0.002	0.009 ± 0.002	0.194 ± 0.029
Métodos		Conjunto de dados Seq			
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-OnlyLabels	0.042 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.097 ± 0.003	0.033 ± 0.003	0.015 ± 0.001	0.005 ± 0.001	0.012 ± 0.004
Bp-True	0.097 ± 0.004	0.034 ± 0.004	0.015 ± 0.001	0.005 ± 0.001	0.013 ± 0.005
Bp-NoLabels	0.099 ± 0.005	0.034 ± 0.004	0.015 ± 0.001	0.005 ± 0.001	0.018 ± 0.011

Tabela A.2: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados estruturados como árvores, utilizando o algoritmo *Resilient Back-propagation*

Conjunto de dados Celleycycle					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.045 ± 0.002	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.085 ± 0.004	0.037 ± 0.002	0.023 ± 0.002	0.009 ± 0.003	0.021 ± 0.012
Rp-True	0.080 ± 0.005	0.032 ± 0.001	0.019 ± 0.003	0.008 ± 0.003	0.018 ± 0.011
Rp-NoLabels	0.082 ± 0.004	0.036 ± 0.004	0.021 ± 0.003	0.006 ± 0.001	0.012 ± 0.010
Conjunto de dados Church					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.042 ± 0.003	0.014 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.056 ± 0.002	0.022 ± 0.001	0.012 ± 0.001	0.006 ± 0.002	0.027 ± 0.020
Rp-True	0.052 ± 0.002	0.019 ± 0.001	0.010 ± 0.001	0.004 ± 0.001	0.002 ± 0.001
Rp-NoLabels	0.053 ± 0.003	0.020 ± 0.001	0.010 ± 0.001	0.004 ± 0.001	0.002 ± 0.001
Conjunto de dados Derisi					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.044 ± 0.002	0.015 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.063 ± 0.002	0.024 ± 0.001	0.012 ± 0.001	0.007 ± 0.003	0.038 ± 0.036
Rp-True	0.063 ± 0.002	0.023 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.005 ± 0.005
Rp-NoLabels	0.062 ± 0.002	0.024 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.004 ± 0.002
Conjunto de dados Eisen					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.051 ± 0.002	0.018 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.119 ± 0.005	0.057 ± 0.006	0.038 ± 0.004	0.018 ± 0.004	0.024 ± 0.034
Rp-True	0.112 ± 0.005	0.048 ± 0.003	0.030 ± 0.003	0.010 ± 0.003	0.021 ± 0.052
Rp-NoLabels	0.115 ± 0.004	0.053 ± 0.003	0.030 ± 0.002	0.012 ± 0.003	0.009 ± 0.013
Conjunto de dados Gasch1					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.046 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.125 ± 0.004	0.062 ± 0.003	0.046 ± 0.004	0.034 ± 0.004	0.217 ± 0.083
Rp-True	0.122 ± 0.005	0.061 ± 0.002	0.047 ± 0.005	0.037 ± 0.005	0.190 ± 0.056
Rp-NoLabels	0.123 ± 0.004	0.059 ± 0.004	0.038 ± 0.005	0.031 ± 0.005	0.176 ± 0.076
Conjunto de dados Gasch2					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.048 ± 0.005	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.093 ± 0.002	0.043 ± 0.002	0.023 ± 0.003	0.012 ± 0.004	0.033 ± 0.029
Rp-True	0.084 ± 0.005	0.031 ± 0.002	0.014 ± 0.001	0.006 ± 0.001	0.037 ± 0.026
Rp-NoLabels	0.091 ± 0.003	0.040 ± 0.002	0.017 ± 0.002	0.007 ± 0.001	0.022 ± 0.018
Conjunto de dados Pheno					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.041 ± 0.001	0.015 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.051 ± 0.002	0.019 ± 0.002	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-True	0.050 ± 0.002	0.018 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-NoLabels	0.052 ± 0.002	0.019 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Conjunto de dados Spo					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.042 ± 0.002	0.014 ± 0.001	0.008 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.061 ± 0.001	0.024 ± 0.001	0.014 ± 0.002	0.007 ± 0.002	0.007 ± 0.008
Rp-True	0.061 ± 0.001	0.022 ± 0.001	0.012 ± 0.001	0.006 ± 0.002	0.011 ± 0.013
Rp-NoLabels	0.061 ± 0.002	0.025 ± 0.001	0.013 ± 0.001	0.006 ± 0.001	0.003 ± 0.001
Conjunto de dados Expr					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.051 ± 0.001	0.016 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.126 ± 0.005	0.072 ± 0.006	0.047 ± 0.004	0.038 ± 0.003	0.265 ± 0.084
Rp-True	0.129 ± 0.006	0.072 ± 0.003	0.052 ± 0.005	0.042 ± 0.007	0.208 ± 0.083
Rp-NoLabels	0.125 ± 0.007	0.068 ± 0.005	0.051 ± 0.002	0.042 ± 0.007	0.261 ± 0.079
Conjunto de dados Seq					
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Rp-OnlyLabels	0.051 ± 0.002	0.015 ± 0.001	0.009 ± 0.001	0.003 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.120 ± 0.005	0.082 ± 0.002	0.045 ± 0.002	0.039 ± 0.010	0.052 ± 0.056
Rp-True	0.117 ± 0.004	0.081 ± 0.006	0.046 ± 0.007	0.034 ± 0.009	0.053 ± 0.049
Rp-NoLabels	0.119 ± 0.005	0.078 ± 0.003	0.042 ± 0.005	0.033 ± 0.016	0.051 ± 0.044

Tabela A.3: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi, Eisen e Gaschl estruturados como DAGs, utilizando o algoritmo *Back-propagation*

Métodos		Conjunto de dados Celcycle				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.053 ± 0.002	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.080 ± 0.001	0.040 ± 0.002	0.025 ± 0.001	0.017 ± 0.001	0.010 ± 0.001	0.006 ± 0.001
Bp-True	0.074 ± 0.002	0.035 ± 0.001	0.021 ± 0.001	0.013 ± 0.001	0.006 ± 0.001	0.004 ± 0.001
Bp-NoLabels	0.078 ± 0.001	0.038 ± 0.001	0.021 ± 0.001	0.009 ± 0.001	0.004 ± 0.001	0.004 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.006 ± 0.002	0.007 ± 0.002	0.005 ± 0.001	0.007 ± 0.001	0.015 ± 0.008	0.016 ± 0.012
Bp-True	0.004 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.006 ± 0.001	0.006 ± 0.002	0.008 ± 0.008
Bp-NoLabels	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.007 ± 0.003	0.007 ± 0.003
Métodos		Conjunto de dados Church				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.051 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.063 ± 0.001	0.028 ± 0.001	0.016 ± 0.001	0.009 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Bp-True	0.059 ± 0.001	0.026 ± 0.001	0.014 ± 0.002	0.008 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Bp-NoLabels	0.064 ± 0.001	0.026 ± 0.001	0.012 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.002	0.008 ± 0.001
Bp-True	0.004 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.006 ± 0.001	0.005 ± 0.001	0.003 ± 0.002
Bp-NoLabels	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.004 ± 0.005
Métodos		Conjunto de dados Derisi				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.051 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.065 ± 0.001	0.030 ± 0.001	0.018 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Bp-True	0.061 ± 0.001	0.027 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Bp-NoLabels	0.066 ± 0.001	0.030 ± 0.001	0.017 ± 0.001	0.009 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.005 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.012 ± 0.002	0.006 ± 0.001
Bp-True	0.004 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.004 ± 0.003
Bp-NoLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.004 ± 0.001
Métodos		Conjunto de dados Eisen				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.061 ± 0.001	0.025 ± 0.001	0.013 ± 0.001	0.008 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Bp-Predicted	0.098 ± 0.002	0.055 ± 0.002	0.039 ± 0.001	0.028 ± 0.002	0.017 ± 0.002	0.013 ± 0.001
Bp-True	0.092 ± 0.001	0.047 ± 0.001	0.031 ± 0.001	0.021 ± 0.001	0.010 ± 0.001	0.007 ± 0.001
Bp-NoLabels	0.097 ± 0.001	0.052 ± 0.002	0.032 ± 0.001	0.016 ± 0.002	0.006 ± 0.001	0.005 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.004 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.002 ± 0.001
Bp-Predicted	0.014 ± 0.002	0.017 ± 0.003	0.009 ± 0.002	0.011 ± 0.002	0.012 ± 0.002	0.005 ± 0.002
Bp-True	0.008 ± 0.001	0.008 ± 0.002	0.007 ± 0.001	0.010 ± 0.001	0.013 ± 0.001	0.007 ± 0.003
Bp-NoLabels	0.006 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.008 ± 0.002	0.012 ± 0.003	0.011 ± 0.008
Métodos		Conjunto de dados Gaschl				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.059 ± 0.004	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.088 ± 0.002	0.048 ± 0.003	0.029 ± 0.003	0.019 ± 0.003	0.011 ± 0.003	0.008 ± 0.002
Bp-True	0.085 ± 0.001	0.043 ± 0.001	0.026 ± 0.001	0.017 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
Bp-NoLabels	0.088 ± 0.001	0.047 ± 0.002	0.027 ± 0.001	0.016 ± 0.001	0.007 ± 0.001	0.005 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.009 ± 0.004	0.009 ± 0.003	0.012 ± 0.006	0.009 ± 0.002	0.005 ± 0.001	0.001 ± 0.001
Bp-True	0.007 ± 0.002	0.007 ± 0.002	0.009 ± 0.004	0.007 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-NoLabels	0.005 ± 0.001	0.005 ± 0.001	0.007 ± 0.001	0.008 ± 0.001	0.005 ± 0.001	0.002 ± 0.001

Tabela A.4: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como DAGs, utilizando o algoritmo *Back-propagation*

Conjunto de dados Gasch2						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.055 ± 0.004	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.080 ± 0.001	0.042 ± 0.002	0.025 ± 0.001	0.018 ± 0.001	0.010 ± 0.001	0.008 ± 0.001
Bp-True	0.075 ± 0.001	0.037 ± 0.001	0.022 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.006 ± 0.001
Bp-NoLabels	0.080 ± 0.001	0.040 ± 0.001	0.020 ± 0.001	0.008 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.008 ± 0.001	0.006 ± 0.001	0.008 ± 0.001	0.011 ± 0.002	0.009 ± 0.001	0.007 ± 0.002
Bp-True	0.006 ± 0.001	0.004 ± 0.001	0.006 ± 0.001	0.010 ± 0.002	0.005 ± 0.001	0.002 ± 0.001
Bp-NoLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.002	0.006 ± 0.001	0.005 ± 0.001	0.003 ± 0.001
Conjunto de dados Pheno						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.057 ± 0.001	0.026 ± 0.001	0.013 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Bp-Predicted	0.058 ± 0.001	0.026 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Bp-True	0.060 ± 0.001	0.028 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Bp-NoLabels	0.060 ± 0.001	0.027 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.003 ± 0.001
Bp-True	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.003 ± 0.001
Bp-NoLabels	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.006 ± 0.001	0.003 ± 0.001
Conjunto de dados Spo						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.051 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.067 ± 0.001	0.034 ± 0.001	0.022 ± 0.001	0.013 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
Bp-True	0.062 ± 0.001	0.029 ± 0.001	0.017 ± 0.001	0.010 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Bp-NoLabels	0.067 ± 0.001	0.033 ± 0.001	0.018 ± 0.001	0.008 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.008 ± 0.001	0.008 ± 0.002	0.005 ± 0.001	0.006 ± 0.001	0.005 ± 0.001	0.002 ± 0.002
Bp-True	0.005 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Bp-NoLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.002 ± 0.001
Conjunto de dados Expr						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.056 ± 0.002	0.022 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.091 ± 0.002	0.051 ± 0.004	0.032 ± 0.002	0.022 ± 0.003	0.014 ± 0.004	0.010 ± 0.002
Bp-True	0.089 ± 0.001	0.047 ± 0.002	0.029 ± 0.002	0.020 ± 0.001	0.011 ± 0.001	0.008 ± 0.001
Bp-NoLabels	0.092 ± 0.002	0.050 ± 0.002	0.030 ± 0.002	0.019 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.010 ± 0.002	0.011 ± 0.003	0.010 ± 0.003	0.010 ± 0.003	0.010 ± 0.003	0.004 ± 0.004
Bp-True	0.009 ± 0.001	0.009 ± 0.002	0.008 ± 0.003	0.009 ± 0.001	0.009 ± 0.003	0.005 ± 0.006
Bp-NoLabels	0.008 ± 0.001	0.009 ± 0.002	0.008 ± 0.001	0.009 ± 0.001	0.008 ± 0.001	0.005 ± 0.002
Conjunto de dados Seq						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-OnlyLabels	0.060 ± 0.004	0.022 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Bp-Predicted	0.104 ± 0.003	0.050 ± 0.003	0.027 ± 0.003	0.019 ± 0.002	0.007 ± 0.001	0.006 ± 0.001
Bp-True	0.101 ± 0.003	0.048 ± 0.002	0.025 ± 0.003	0.018 ± 0.001	0.007 ± 0.001	0.005 ± 0.001
Bp-NoLabels	0.106 ± 0.003	0.051 ± 0.002	0.025 ± 0.002	0.017 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Bp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Bp-Predicted	0.005 ± 0.001	0.005 ± 0.001	0.008 ± 0.002	0.010 ± 0.001	0.011 ± 0.002	0.005 ± 0.001
Bp-True	0.006 ± 0.001	0.005 ± 0.001	0.007 ± 0.001	0.010 ± 0.002	0.010 ± 0.002	0.005 ± 0.001
Bp-NoLabels	0.005 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.009 ± 0.001	0.011 ± 0.002	0.006 ± 0.001

Tabela A.5: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Celcycle, Church, Derisi, Eisen e Gaschl estruturados como DAGs, utilizando o algoritmo *Resilient Back-propagation*

Métodos		Conjunto de dados Celcycle				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.055 ± 0.002	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.084 ± 0.002	0.058 ± 0.003	0.039 ± 0.002	0.024 ± 0.002	0.017 ± 0.003	0.013 ± 0.003
Rp-True	0.081 ± 0.001	0.051 ± 0.004	0.029 ± 0.003	0.018 ± 0.002	0.013 ± 0.002	0.010 ± 0.002
Rp-NoLabels	0.082 ± 0.002	0.045 ± 0.004	0.028 ± 0.002	0.020 ± 0.001	0.014 ± 0.002	0.010 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.014 ± 0.002	0.013 ± 0.002	0.013 ± 0.002	0.008 ± 0.001	0.007 ± 0.001	0.005 ± 0.002
Rp-True	0.011 ± 0.002	0.010 ± 0.002	0.012 ± 0.004	0.008 ± 0.003	0.007 ± 0.001	0.004 ± 0.003
Rp-NoLabels	0.011 ± 0.002	0.010 ± 0.002	0.010 ± 0.003	0.007 ± 0.001	0.009 ± 0.001	0.007 ± 0.008
Métodos		Conjunto de dados Church				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.053 ± 0.001	0.022 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.065 ± 0.001	0.035 ± 0.001	0.019 ± 0.001	0.012 ± 0.001	0.008 ± 0.001	0.007 ± 0.001
Rp-True	0.063 ± 0.001	0.032 ± 0.001	0.018 ± 0.001	0.011 ± 0.001	0.007 ± 0.001	0.007 ± 0.001
Rp-NoLabels	0.063 ± 0.001	0.032 ± 0.002	0.016 ± 0.001	0.010 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.008 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.007 ± 0.001	0.007 ± 0.001	0.004 ± 0.001
Rp-True	0.008 ± 0.002	0.006 ± 0.001	0.006 ± 0.001	0.006 ± 0.001	0.006 ± 0.002	0.002 ± 0.001
Rp-NoLabels	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.002	0.007 ± 0.002	0.006 ± 0.001	0.002 ± 0.001
Métodos		Conjunto de dados Derisi				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.055 ± 0.004	0.022 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.067 ± 0.001	0.036 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Rp-True	0.065 ± 0.001	0.035 ± 0.002	0.020 ± 0.002	0.011 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Rp-NoLabels	0.064 ± 0.001	0.033 ± 0.002	0.020 ± 0.001	0.012 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.006 ± 0.001	0.006 ± 0.001	0.008 ± 0.001	0.007 ± 0.001	0.012 ± 0.003	0.006 ± 0.002
Rp-True	0.005 ± 0.001	0.006 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.012 ± 0.002	0.006 ± 0.003
Rp-NoLabels	0.007 ± 0.001	0.006 ± 0.001	0.008 ± 0.001	0.008 ± 0.001	0.016 ± 0.006	0.009 ± 0.004
Métodos		Conjunto de dados Eisen				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.064 ± 0.001	0.026 ± 0.001	0.013 ± 0.001	0.008 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Rp-Predicted	0.108 ± 0.003	0.082 ± 0.003	0.055 ± 0.003	0.040 ± 0.003	0.030 ± 0.006	0.023 ± 0.003
Rp-True	0.104 ± 0.002	0.076 ± 0.002	0.049 ± 0.004	0.034 ± 0.004	0.025 ± 0.002	0.021 ± 0.002
Rp-NoLabels	0.102 ± 0.002	0.068 ± 0.004	0.043 ± 0.002	0.033 ± 0.001	0.026 ± 0.003	0.019 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.004 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.023 ± 0.004	0.026 ± 0.003	0.026 ± 0.003	0.018 ± 0.007	0.031 ± 0.018	0.022 ± 0.015
Rp-True	0.021 ± 0.002	0.023 ± 0.004	0.024 ± 0.005	0.019 ± 0.009	0.037 ± 0.018	0.063 ± 0.058
Rp-NoLabels	0.018 ± 0.001	0.020 ± 0.002	0.018 ± 0.003	0.015 ± 0.002	0.031 ± 0.014	0.040 ± 0.027
Métodos		Conjunto de dados Gaschl				
	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.058 ± 0.003	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.100 ± 0.002	0.073 ± 0.003	0.038 ± 0.005	0.023 ± 0.004	0.020 ± 0.005	0.017 ± 0.003
Rp-True	0.097 ± 0.002	0.064 ± 0.003	0.036 ± 0.002	0.022 ± 0.002	0.018 ± 0.002	0.017 ± 0.003
Rp-NoLabels	0.097 ± 0.003	0.069 ± 0.003	0.046 ± 0.001	0.032 ± 0.002	0.029 ± 0.002	0.024 ± 0.003
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.023 ± 0.004	0.023 ± 0.005	0.023 ± 0.003	0.014 ± 0.003	0.013 ± 0.006	0.005 ± 0.002
Rp-True	0.020 ± 0.003	0.019 ± 0.004	0.023 ± 0.003	0.017 ± 0.003	0.011 ± 0.003	0.005 ± 0.001
Rp-NoLabels	0.027 ± 0.004	0.019 ± 0.002	0.025 ± 0.003	0.018 ± 0.005	0.018 ± 0.016	0.005 ± 0.002

Tabela A.6: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como DAGs, utilizando o algoritmo *Resilient Back-propagation*

Conjunto de dados Gasch2						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.056 ± 0.002	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.086 ± 0.002	0.065 ± 0.002	0.039 ± 0.001	0.023 ± 0.002	0.017 ± 0.003	0.016 ± 0.003
Rp-True	0.082 ± 0.003	0.060 ± 0.003	0.034 ± 0.002	0.020 ± 0.002	0.015 ± 0.003	0.012 ± 0.001
Rp-NoLabels	0.082 ± 0.002	0.057 ± 0.006	0.031 ± 0.002	0.018 ± 0.001	0.014 ± 0.002	0.011 ± 0.002
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.016 ± 0.002	0.012 ± 0.001	0.014 ± 0.003	0.011 ± 0.002	0.009 ± 0.002	0.002 ± 0.002
Rp-True	0.013 ± 0.002	0.010 ± 0.001	0.016 ± 0.001	0.011 ± 0.002	0.010 ± 0.004	0.004 ± 0.003
Rp-NoLabels	0.011 ± 0.001	0.008 ± 0.002	0.010 ± 0.002	0.012 ± 0.002	0.009 ± 0.003	0.003 ± 0.002
Conjunto de dados Pheno						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.058 ± 0.001	0.026 ± 0.001	0.013 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Rp-Predicted	0.064 ± 0.001	0.031 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Rp-True	0.063 ± 0.001	0.030 ± 0.001	0.016 ± 0.001	0.009 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Rp-NoLabels	0.063 ± 0.001	0.029 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.005 ± 0.001	0.006 ± 0.001	0.009 ± 0.001	0.010 ± 0.002	0.034 ± 0.019	0.009 ± 0.002
Rp-True	0.005 ± 0.001	0.006 ± 0.001	0.009 ± 0.001	0.013 ± 0.004	0.057 ± 0.036	0.008 ± 0.001
Rp-NoLabels	0.005 ± 0.001	0.005 ± 0.001	0.008 ± 0.001	0.010 ± 0.002	0.024 ± 0.014	0.008 ± 0.001
Conjunto de dados Spo						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.051 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.067 ± 0.001	0.034 ± 0.001	0.021 ± 0.001	0.012 ± 0.001	0.009 ± 0.001	0.008 ± 0.001
Rp-True	0.065 ± 0.001	0.032 ± 0.001	0.020 ± 0.002	0.012 ± 0.001	0.009 ± 0.001	0.008 ± 0.001
Rp-NoLabels	0.065 ± 0.001	0.033 ± 0.001	0.020 ± 0.001	0.014 ± 0.001	0.012 ± 0.001	0.009 ± 0.001
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.009 ± 0.001	0.010 ± 0.001	0.008 ± 0.003	0.008 ± 0.001	0.007 ± 0.002	0.008 ± 0.008
Rp-True	0.010 ± 0.001	0.010 ± 0.002	0.008 ± 0.002	0.009 ± 0.002	0.008 ± 0.003	0.006 ± 0.005
Rp-NoLabels	0.010 ± 0.002	0.010 ± 0.002	0.008 ± 0.001	0.007 ± 0.001	0.007 ± 0.003	0.014 ± 0.018
Conjunto de dados Expr						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.064 ± 0.004	0.022 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.098 ± 0.004	0.075 ± 0.003	0.044 ± 0.002	0.030 ± 0.001	0.027 ± 0.003	0.025 ± 0.002
Rp-True	0.098 ± 0.005	0.065 ± 0.009	0.037 ± 0.004	0.029 ± 0.003	0.026 ± 0.004	0.024 ± 0.004
Rp-NoLabels	0.100 ± 0.005	0.075 ± 0.005	0.048 ± 0.003	0.038 ± 0.002	0.041 ± 0.003	0.035 ± 0.002
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.030 ± 0.002	0.026 ± 0.003	0.029 ± 0.005	0.014 ± 0.003	0.027 ± 0.009	0.008 ± 0.003
Rp-True	0.024 ± 0.004	0.026 ± 0.004	0.031 ± 0.007	0.014 ± 0.005	0.019 ± 0.009	0.005 ± 0.001
Rp-NoLabels	0.035 ± 0.005	0.027 ± 0.004	0.030 ± 0.005	0.016 ± 0.006	0.024 ± 0.013	0.006 ± 0.003
Conjunto de dados Seq						
Métodos	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Rp-OnlyLabels	0.078 ± 0.005	0.025 ± 0.003	0.011 ± 0.002	0.006 ± 0.001	0.004 ± 0.001	0.003 ± 0.001
Rp-Predicted	0.129 ± 0.007	0.082 ± 0.004	0.059 ± 0.002	0.044 ± 0.003	0.033 ± 0.006	0.036 ± 0.003
Rp-True	0.124 ± 0.006	0.074 ± 0.006	0.045 ± 0.004	0.031 ± 0.002	0.019 ± 0.003	0.024 ± 0.003
Rp-NoLabels	0.129 ± 0.006	0.073 ± 0.008	0.055 ± 0.006	0.046 ± 0.003	0.036 ± 0.003	0.040 ± 0.003
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13
Rp-OnlyLabels	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001	0.004 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.028 ± 0.006	0.026 ± 0.006	0.023 ± 0.006	0.066 ± 0.014	0.194 ± 0.030	0.174 ± 0.067
Rp-True	0.020 ± 0.005	0.020 ± 0.004	0.023 ± 0.005	0.076 ± 0.007	0.215 ± 0.023	0.168 ± 0.001
Rp-NoLabels	0.037 ± 0.005	0.028 ± 0.006	0.026 ± 0.003	0.072 ± 0.014	0.206 ± 0.024	0.172 ± 0.011

Resultados por Nível das Variações Bp-Predicted, Rp-Predicted e Métodos da Literatura

Este Apêndice apresenta uma comparação das variações Bp-Predicted e Rp-Predicted, e também de todos os métodos da literatura, quanto aos desempenhos obtidos em cada nível. Esses resultados são apresentados nas Tabelas [B.1](#), [B.2](#), [B.3](#), [B.4](#) e [B.5](#). Pode-se observar que as variações Bp-Predicted e Rp-Predicted obtiveram os melhores resultados na maioria dos níveis, considerando tanto as hierarquias estruturadas como árvores quanto as hierarquias estruturadas como grafos.

Observando os resultados no primeiro nível de cada hierarquia, tem-se que os melhores resultados foram obtidos pelas variações do método [HMC-LMLP](#). Isso mostra como as redes neurais obtiveram os melhores resultados na maioria das classes mais frequentes nos conjuntos de dados. Ainda, se observados os desempenhos dos métodos nos níveis mais profundos, apesar das diferenças entre os resultados não serem muito grandes, é possível observar que os resultados das variações do método [HMC-LMLP](#) foram ligeiramente melhores. Isso mostra que o método [HMC-LMLP](#) foi capaz de superar os métodos da literatura em classes pouco frequentes nos conjuntos de dados.

Tabela B.1: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Cellcycle, Church, Derisi, Eisen e Gaschl estruturados como árvores, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura

Métodos		Conjunto de dados Cellcycle				
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.243 ± 0.002	0.081 ± 0.004	0.030 ± 0.001	0.017 ± 0.001	0.004 ± 0.001	0.011 ± 0.004
Rp-Predicted	0.239 ± 0.002	0.085 ± 0.004	0.037 ± 0.002	0.023 ± 0.002	0.009 ± 0.003	0.021 ± 0.012
Clus-HMC	0.184	0.059	0.027	0.015	0.006	0.002
Clus-HSC	0.189	0.065	0.030	0.019	0.011	0.005
Clus-SC	0.189	0.065	0.028	0.022	0.013	0.171
hmAnt-Miner	0.174 ± 0.005	0.055 ± 0.002	0.024 ± 0.001	0.014 ± 0.001	0.006 ± 0.001	0.010 ± 0.019
Métodos		Conjunto de dados Church				
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.181 ± 0.001	0.051 ± 0.001	0.018 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.173 ± 0.001	0.056 ± 0.002	0.022 ± 0.001	0.012 ± 0.001	0.006 ± 0.002	0.027 ± 0.020
Clus-HMC	0.173	0.054	0.022	0.012	0.006	0.025
Clus-HSC	0.166	0.052	0.023	0.014	0.007	0.002
Clus-SC	0.166	0.055	0.026	0.013	0.010	0.085
hmAnt-Miner	0.161 ± 0.001	0.049 ± 0.001	0.020 ± 0.001	0.012 ± 0.001	0.005 ± 0.001	0.003 ± 0.001
Métodos		Conjunto de dados Derisi				
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.203 ± 0.002	0.063 ± 0.002	0.022 ± 0.001	0.011 ± 0.001	0.004 ± 0.001	0.002 ± 0.001
Rp-Predicted	0.198 ± 0.002	0.063 ± 0.002	0.024 ± 0.001	0.012 ± 0.001	0.007 ± 0.003	0.038 ± 0.036
Clus-HMC	0.170	0.054	0.020	0.010	0.004	0.002
Clus-HSC	0.165	0.058	0.022	0.013	0.004	0.004
Clus-SC	0.165	0.053	0.022	0.012	0.005	0.002
hmAnt-Miner	0.180 ± 0.003	0.059 ± 0.001	0.024 ± 0.001	0.013 ± 0.001	0.007 ± 0.002	0.006 ± 0.005
Métodos		Conjunto de dados Eisen				
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.292 ± 0.002	0.113 ± 0.005	0.043 ± 0.003	0.025 ± 0.002	0.006 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.291 ± 0.003	0.119 ± 0.005	0.057 ± 0.006	0.038 ± 0.004	0.018 ± 0.004	0.024 ± 0.034
Clus-HMC	0.237	0.084	0.038	0.023	0.006	0.001
Clus-HSC	0.239	0.082	0.042	0.035	0.012	0.001
Clus-SC	0.239	0.092	0.049	0.029	0.013	0.004
hmAnt-Miner	0.209 ± 0.010	0.069 ± 0.004	0.032 ± 0.002	0.019 ± 0.003	0.008 ± 0.002	0.006 ± 0.006
Métodos		Conjunto de dados Gaschl				
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.269 ± 0.002	0.115 ± 0.003	0.038 ± 0.002	0.019 ± 0.001	0.008 ± 0.001	0.172 ± 0.026
Rp-Predicted	0.266 ± 0.002	0.125 ± 0.004	0.062 ± 0.003	0.046 ± 0.004	0.034 ± 0.004	0.217 ± 0.083
Clus-HMC	0.211	0.072	0.029	0.013	0.005	0.002
Clus-HSC	0.193	0.076	0.036	0.031	0.028	0.047
Clus-SC	0.193	0.083	0.036	0.022	0.035	0.201
hmAnt-Miner	0.205 ± 0.005	0.068 ± 0.002	0.029 ± 0.001	0.015 ± 0.001	0.007 ± 0.002	0.005 ± 0.004

Tabela B.2: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch2, Pheno, Spo, Expr e Seq estruturados como árvores, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura

Métodos	Conjunto de dados Gasch2					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.234 ± 0.002	0.093 ± 0.002	0.031 ± 0.001	0.016 ± 0.001	0.006 ± 0.001	0.083 ± 0.037
Rp-Predicted	0.229 ± 0.003	0.093 ± 0.002	0.043 ± 0.002	0.023 ± 0.003	0.012 ± 0.004	0.033 ± 0.029
Clus-HMC	0.204	0.071	0.027	0.013	0.005	0.002
Clus-HSC	0.189	0.070	0.033	0.021	0.018	0.276
Clus-SC	0.189	0.069	0.030	0.019	0.010	0.005
hmAnt-Miner	0.188 ± 0.004	0.062 ± 0.002	0.026 ± 0.001	0.013 ± 0.001	0.006 ± 0.001	0.005 ± 0.004
Métodos	Conjunto de dados Pheno					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.175 ± 0.005	0.052 ± 0.002	0.016 ± 0.001	0.010 ± 0.001	0.003 ± 0.001	0.001 ± 0.001
Rp-Predicted	0.167 ± 0.003	0.051 ± 0.002	0.019 ± 0.002	0.011 ± 0.001	0.004 ± 0.001	0.001 ± 0.001
Clus-HMC	0.165	0.054	0.023	0.013	0.004	0.001
Clus-HSC	0.168	0.055	0.022	0.018	0.004	0.002
Clus-SC	0.168	0.058	0.023	0.013	0.005	0.001
hmAnt-Miner	0.163 ± 0.002	0.051 ± 0.001	0.019 ± 0.001	0.013 ± 0.001	0.004 ± 0.001	0.002 ± 0.001
Métodos	Conjunto de dados Spo					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.207 ± 0.002	0.062 ± 0.001	0.023 ± 0.001	0.012 ± 0.001	0.004 ± 0.001	0.009 ± 0.004
Rp-Predicted	0.197 ± 0.003	0.061 ± 0.001	0.024 ± 0.001	0.014 ± 0.002	0.007 ± 0.002	0.007 ± 0.008
Clus-HMC	0.182	0.057	0.024	0.012	0.005	0.007
Clus-HSC	0.178	0.060	0.028	0.022	0.032	0.002
Clus-SC	0.178	0.060	0.024	0.019	0.014	0.032
hmAnt-Miner	0.184 ± 0.003	0.060 ± 0.001	0.025 ± 0.001	0.015 ± 0.001	0.006 ± 0.001	0.004 ± 0.002
Métodos	Conjunto de dados Expr					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.287 ± 0.002	0.128 ± 0.004	0.047 ± 0.003	0.025 ± 0.001	0.010 ± 0.001	0.203 ± 0.023
Rp-Predicted	0.279 ± 0.005	0.126 ± 0.005	0.072 ± 0.006	0.047 ± 0.004	0.038 ± 0.003	0.265 ± 0.084
Clus-HMC	0.216	0.078	0.034	0.016	0.005	0.002
Clus-HSC	0.211	0.079	0.049	0.035	0.032	0.015
Clus-SC	0.211	0.085	0.041	0.032	0.020	0.020
hmAnt-Miner	0.208 ± 0.005	0.069 ± 0.001	0.031 ± 0.001	0.018 ± 0.001	0.008 ± 0.001	0.038 ± 0.039
Métodos	Conjunto de dados Seq					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Bp-Predicted	0.312 ± 0.002	0.097 ± 0.003	0.033 ± 0.003	0.015 ± 0.001	0.005 ± 0.001	0.012 ± 0.004
Rp-Predicted	0.318 ± 0.002	0.120 ± 0.005	0.082 ± 0.002	0.045 ± 0.002	0.039 ± 0.010	0.052 ± 0.056
Clus-HMC	0.246	0.070	0.033	0.017	0.008	0.002
Clus-HSC	0.224	0.062	0.034	0.027	0.021	0.010
Clus-SC	0.224	0.071	0.032	0.026	0.014	0.025
hmAnt-Miner	0.234 ± 0.005	0.063 ± 0.003	0.027 ± 0.001	0.016 ± 0.001	0.008 ± 0.001	0.006 ± 0.001

Tabela B.3: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Cellcycle, Church, Derisi e Eisen estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura

Conjunto de dados Cellcycle							
Métodos	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.226 ± 0.002	0.080 ± 0.001	0.040 ± 0.002	0.025 ± 0.001	0.017 ± 0.001	0.010 ± 0.001	0.006 ± 0.001
Rp-Predicted	0.229 ± 0.003	0.084 ± 0.002	0.058 ± 0.003	0.039 ± 0.002	0.024 ± 0.002	0.017 ± 0.003	0.013 ± 0.003
Clus-HMC	0.203	0.068	0.033	0.020	0.013	0.009	0.007
Clus-HSC	0.206	0.079	0.043	0.027	0.017	0.008	0.005
Clus-SC	0.206	0.073	0.037	0.020	0.014	0.011	0.009
hmAnt-Miner	0.198 ± 0.001	0.065 ± 0.001	0.032 ± 0.002	0.019 ± 0.001	0.013 ± 0.001	0.010 ± 0.001	0.008 ± 0.001
Conjunto de dados Church							
Métodos	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.196 ± 0.002	0.063 ± 0.001	0.028 ± 0.001	0.016 ± 0.001	0.009 ± 0.001	0.005 ± 0.001	0.004 ± 0.001
Rp-Predicted	0.198 ± 0.002	0.065 ± 0.001	0.035 ± 0.001	0.019 ± 0.001	0.012 ± 0.001	0.008 ± 0.001	0.007 ± 0.001
Clus-HMC	0.192	0.059	0.026	0.014	0.009	0.005	0.005
Clus-HSC	0.192	0.061	0.030	0.016	0.009	0.006	0.004
Clus-SC	0.192	0.061	0.030	0.019	0.012	0.008	0.006
hmAnt-Miner	0.191 ± 0.003	0.062 ± 0.001	0.032 ± 0.002	0.018 ± 0.001	0.010 ± 0.001	0.008 ± 0.001	0.006 ± 0.001
Conjunto de dados Derisi							
Métodos	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.203 ± 0.001	0.065 ± 0.001	0.030 ± 0.001	0.018 ± 0.001	0.011 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Rp-Predicted	0.203 ± 0.001	0.067 ± 0.001	0.036 ± 0.001	0.021 ± 0.001	0.011 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Clus-HMC	0.199	0.064	0.030	0.019	0.011	0.007	0.006
Clus-HSC	0.190	0.061	0.028	0.018	0.009	0.005	0.004
Clus-SC	0.190	0.061	0.029	0.017	0.012	0.009	0.008
hmAnt-Miner	0.192 ± 0.002	0.063 ± 0.001	0.030 ± 0.001	0.020 ± 0.001	0.012 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
Conjunto de dados Eisen							
Métodos	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.257 ± 0.003	0.098 ± 0.002	0.055 ± 0.002	0.039 ± 0.001	0.028 ± 0.002	0.017 ± 0.002	0.013 ± 0.001
Rp-Predicted	0.259 ± 0.004	0.108 ± 0.003	0.082 ± 0.003	0.055 ± 0.003	0.040 ± 0.003	0.030 ± 0.006	0.023 ± 0.003
Clus-HMC	0.238	0.089	0.053	0.034	0.023	0.019	0.012
Clus-HSC	0.243	0.089	0.056	0.037	0.023	0.016	0.013
Clus-SC	0.243	0.087	0.051	0.038	0.024	0.018	0.011
hmAnt-Miner	0.230 ± 0.004	0.081 ± 0.002	0.044 ± 0.002	0.028 ± 0.002	0.019 ± 0.001	0.014 ± 0.001	0.012 ± 0.002
Conjunto de dados Eisen (continuação)							
Métodos	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.014 ± 0.002	0.017 ± 0.003	0.009 ± 0.002	0.011 ± 0.002	0.012 ± 0.002	0.005 ± 0.002	
Rp-Predicted	0.023 ± 0.004	0.026 ± 0.003	0.026 ± 0.003	0.018 ± 0.007	0.031 ± 0.018	0.022 ± 0.015	
Clus-HMC	0.010	0.009	0.012	0.007	0.006	0.002	
Clus-HSC	0.021	0.009	0.005	0.006	0.007	0.003	
Clus-SC	0.013	0.010	0.008	0.014	0.010	0.043	
hmAnt-Miner	0.011 ± 0.001	0.011 ± 0.001	0.011 ± 0.003	0.017 ± 0.015	0.012 ± 0.003	0.006 ± 0.002	

Tabela B.4: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Gasch1, Gasch2, Pheno e Spo estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura

Métodos		Conjunto de dados Gasch1					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.243 ± 0.003	0.088 ± 0.002	0.048 ± 0.003	0.029 ± 0.003	0.019 ± 0.003	0.011 ± 0.003	0.008 ± 0.002
Rp-Predicted	0.249 ± 0.003	0.100 ± 0.002	0.073 ± 0.003	0.038 ± 0.005	0.023 ± 0.004	0.020 ± 0.005	0.017 ± 0.003
Clus-HMC	0.220	0.076	0.041	0.028	0.018	0.013	0.010
Clus-HSC	0.218	0.079	0.047	0.029	0.018	0.011	0.007
Clus-SC	0.218	0.077	0.043	0.028	0.020	0.015	0.014
hmAnt-Miner	0.209 ± 0.001	0.070 ± 0.001	0.036 ± 0.002	0.023 ± 0.001	0.015 ± 0.001	0.010 ± 0.001	0.009 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.009 ± 0.004	0.009 ± 0.003	0.012 ± 0.006	0.009 ± 0.002	0.005 ± 0.001	0.001 ± 0.001	
Rp-Predicted	0.023 ± 0.004	0.023 ± 0.005	0.023 ± 0.003	0.014 ± 0.003	0.013 ± 0.006	0.005 ± 0.002	
Clus-HMC	0.010	0.010	0.009	0.006	0.005	0.002	
Clus-HSC	0.005	0.005	0.004	0.005	0.005	0.002	
Clus-SC	0.016	0.011	0.013	0.016	0.056	0.001	
hmAnt-Miner	0.008 ± 0.001	0.009 ± 0.001	0.008 ± 0.001	0.009 ± 0.002	0.007 ± 0.003	0.004 ± 0.003	
Métodos		Conjunto de dados Gasch2					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.227 ± 0.001	0.080 ± 0.001	0.042 ± 0.002	0.025 ± 0.001	0.018 ± 0.001	0.010 ± 0.001	0.008 ± 0.001
Rp-Predicted	0.231 ± 0.002	0.086 ± 0.002	0.065 ± 0.002	0.039 ± 0.001	0.023 ± 0.002	0.017 ± 0.003	0.016 ± 0.003
Clus-HMC	0.214	0.072	0.039	0.023	0.014	0.010	0.008
Clus-HSC	0.209	0.077	0.041	0.025	0.016	0.008	0.006
Clus-SC	0.209	0.072	0.038	0.023	0.016	0.013	0.012
hmAnt-Miner	0.205 ± 0.002	0.068 ± 0.001	0.036 ± 0.001	0.022 ± 0.001	0.014 ± 0.001	0.010 ± 0.001	0.008 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.008 ± 0.001	0.006 ± 0.001	0.008 ± 0.001	0.011 ± 0.002	0.009 ± 0.001	0.007 ± 0.002	
Rp-Predicted	0.016 ± 0.002	0.012 ± 0.001	0.014 ± 0.003	0.011 ± 0.002	0.009 ± 0.002	0.002 ± 0.002	
Clus-HMC	0.007	0.006	0.007	0.005	0.005	0.002	
Clus-HSC	0.004	0.006	0.006	0.005	0.005	0.002	
Clus-SC	0.012	0.009	0.007	0.006	0.004	0.001	
hmAnt-Miner	0.007 ± 0.001	0.007 ± 0.001	0.008 ± 0.001	0.007 ± 0.001	0.009 ± 0.002	0.005 ± 0.002	
Métodos		Conjunto de dados Pheno					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.175 ± 0.004	0.058 ± 0.001	0.026 ± 0.001	0.014 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.005 ± 0.001
Rp-Predicted	0.183 ± 0.003	0.064 ± 0.001	0.031 ± 0.001	0.015 ± 0.001	0.009 ± 0.001	0.007 ± 0.001	0.006 ± 0.001
Clus-HMC	0.188	0.069	0.031	0.019	0.012	0.010	0.006
Clus-HSC	0.187	0.066	0.032	0.018	0.010	0.007	0.006
Clus-SC	0.187	0.065	0.033	0.021	0.016	0.012	0.008
hmAnt-Miner	0.180 ± 0.002	0.063 ± 0.001	0.032 ± 0.005	0.019 ± 0.003	0.013 ± 0.003	0.009 ± 0.001	0.008 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.004 ± 0.001	0.004 ± 0.001	0.005 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.003 ± 0.001	
Rp-Predicted	0.005 ± 0.001	0.006 ± 0.001	0.009 ± 0.001	0.010 ± 0.002	0.034 ± 0.019	0.009 ± 0.002	
Clus-HMC	0.006	0.005	0.007	0.010	0.007	0.004	
Clus-HSC	0.005	0.005	0.006	0.006	0.007	0.004	
Clus-SC	0.008	0.006	0.006	0.006	0.010	0.007	
hmAnt-Miner	0.006 ± 0.001	0.007 ± 0.001	0.008 ± 0.001	0.013 ± 0.001	0.029 ± 0.001	0.019 ± 0.001	
Métodos		Conjunto de dados Spo					
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.200 ± 0.002	0.067 ± 0.001	0.034 ± 0.001	0.022 ± 0.001	0.013 ± 0.001	0.009 ± 0.001	0.007 ± 0.001
Rp-Predicted	0.202 ± 0.003	0.067 ± 0.001	0.034 ± 0.001	0.021 ± 0.001	0.012 ± 0.001	0.009 ± 0.001	0.008 ± 0.001
Clus-HMC	0.197	0.066	0.032	0.020	0.013	0.009	0.008
Clus-HSC	0.194	0.063	0.033	0.018	0.013	0.010	0.004
Clus-SC	0.194	0.066	0.031	0.020	0.014	0.011	0.009
hmAnt-Miner	0.194 ± 0.003	0.066 ± 0.002	0.032 ± 0.001	0.020 ± 0.001	0.014 ± 0.001	0.010 ± 0.001	0.009 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.008 ± 0.001	0.008 ± 0.002	0.005 ± 0.001	0.006 ± 0.001	0.005 ± 0.001	0.002 ± 0.002	
Rp-Predicted	0.009 ± 0.001	0.010 ± 0.001	0.008 ± 0.003	0.008 ± 0.001	0.007 ± 0.002	0.008 ± 0.008	
Clus-HMC	0.007	0.006	0.004	0.006	0.005	0.002	
Clus-HSC	0.004	0.003	0.004	0.005	0.005	0.002	
Clus-SC	0.010	0.010	0.004	0.036	0.004	0.001	
hmAnt-Miner	0.007 ± 0.001	0.007 ± 0.001	0.007 ± 0.001	0.007 ± 0.001	0.009 ± 0.003	0.004 ± 0.002	

Tabela B.5: Média das $AU(\overline{PRC})$ obtidas em cada classe, por nível, nos conjuntos de dados Expr e Seq estruturados como DAGs, utilizando os métodos Bp-Predicted, Rp-Predicted e métodos da literatura

Métodos	Conjunto de dados Expr						
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.245 ± 0.002	0.091 ± 0.002	0.051 ± 0.004	0.032 ± 0.002	0.022 ± 0.003	0.014 ± 0.004	0.010 ± 0.002
Rp-Predicted	0.245 ± 0.004	0.098 ± 0.004	0.075 ± 0.003	0.044 ± 0.002	0.030 ± 0.001	0.027 ± 0.003	0.025 ± 0.002
Clus-HMC	0.223	0.077	0.040	0.025	0.016	0.011	0.010
Clus-HSC	0.220	0.077	0.044	0.028	0.021	0.014	0.005
Clus-SC	0.220	0.076	0.047	0.028	0.022	0.019	0.015
hmAnt-Miner	0.207 ± 0.004	0.068 ± 0.002	0.034 ± 0.002	0.021 ± 0.001	0.013 ± 0.001	0.010 ± 0.001	0.007 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.010 ± 0.002	0.011 ± 0.003	0.010 ± 0.003	0.010 ± 0.003	0.010 ± 0.003	0.004 ± 0.004	
Rp-Predicted	0.030 ± 0.002	0.026 ± 0.003	0.029 ± 0.005	0.014 ± 0.003	0.027 ± 0.009	0.008 ± 0.003	
Clus-HMC	0.009	0.007	0.008	0.006	0.005	0.002	
Clus-HSC	0.004	0.005	0.007	0.005	0.005	0.002	
Clus-SC	0.019	0.009	0.007	0.006	0.004	0.001	
hmAnt-Miner	0.006 ± 0.001	0.005 ± 0.001	0.006 ± 0.002	0.008 ± 0.002	0.007 ± 0.003	0.003 ± 0.001	
Métodos	Conjunto de dados Seq						
	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6	Nível 7
Bp-Predicted	0.270 ± 0.002	0.104 ± 0.003	0.050 ± 0.003	0.027 ± 0.003	0.019 ± 0.002	0.007 ± 0.001	0.006 ± 0.001
Rp-Predicted	0.281 ± 0.009	0.129 ± 0.007	0.082 ± 0.004	0.059 ± 0.002	0.044 ± 0.003	0.033 ± 0.006	0.036 ± 0.003
Clus-HMC	0.244	0.090	0.044	0.024	0.018	0.009	0.007
Clus-HSC	0.229	0.099	0.054	0.036	0.032	0.018	0.009
Clus-SC	0.229	0.093	0.054	0.036	0.031	0.028	0.017
hmAnt-Miner	0.232 ± 0.002	0.083 ± 0.002	0.038 ± 0.001	0.021 ± 0.001	0.016 ± 0.001	0.008 ± 0.001	0.007 ± 0.001
	Nível 8	Nível 9	Nível 10	Nível 11	Nível 12	Nível 13	
Bp-Predicted	0.005 ± 0.001	0.005 ± 0.001	0.008 ± 0.002	0.010 ± 0.001	0.011 ± 0.002	0.005 ± 0.001	
Rp-Predicted	0.028 ± 0.006	0.026 ± 0.006	0.023 ± 0.006	0.066 ± 0.014	0.194 ± 0.030	0.174 ± 0.067	
Clus-HMC	0.007	0.010	0.012	0.022	0.078	0.023	
Clus-HSC	0.008	0.006	0.007	0.005	0.005	0.002	
Clus-SC	0.016	0.015	0.010	0.023	0.073	0.025	
hmAnt-Miner	0.006 ± 0.001	0.006 ± 0.001	0.007 ± 0.001	0.008 ± 0.001	0.006 ± 0.001	0.004 ± 0.001	

Resultados das Variações Bp-Predicted, Rp-Predicted e Métodos da Literatura em Classes Específicas

Este Apêndice apresenta uma comparação das variações Bp-Predicted e Rp-Predicted, e também dos métodos na literatura, quanto aos seus desempenhos em classes específicas das hierarquias. Esses resultados são apresentados na Tabelas [C.1](#), [C.2](#), [C.3](#), [C.4](#), [C.5](#), [C.6](#), [C.7](#), [C.8](#), [C.9](#) e [C.10](#). Foram consideradas, em cada nível, as três classes mais frequentes e as três classes menos frequentes nos conjuntos de dados. Isso foi feito para comparar os desempenhos dos métodos em classes com muitos e poucos exemplos positivos.

Nas tabelas apresentadas, são mostradas classes até o quarto nível para hierarquias estruturadas como árvores, e até o sétimo nível para hierarquias estruturadas como grafos. Isso ocorre porque foram consideradas apenas classes cujas frequências nos conjuntos de treinamento fossem maiores do que 5%. Por esse mesmo motivo, alguns níveis hierárquicos mostrados nas tabelas (nível 4 nas hierarquias estruturadas como árvores e níveis 6 e 7 nas hierarquias estruturadas como grafos) não apresentam seis classes (três mais frequentes e três menos frequentes). Nesses casos, as classes mostradas são as únicas com frequências acima de 5% no conjunto de treinamento.

Nos resultados obtidos nas classes específicas apresentadas nas tabelas, pode-se observar que as variações Bp-Predicted e Rp-Predicted obtiveram os melhores resultados na maioria das classes investigadas, mostrando que o método [HMC-LMLP](#) foi capaz de superar os métodos da literatura em classes muito e pouco frequentes. Esse comportamento pode ser observado nas classes tanto das hierarquias estruturadas como árvores quanto das hierarquias estruturadas como grafos.

Tabela C.1: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Cellcycle

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.511 ± 0.005	0.483 ± 0.014	0.402	0.442	0.442	0.383 ± 0.015
	1	14	0.386 ± 0.007	0.375 ± 0.010	0.303	0.303	0.303	0.278 ± 0.012
	1	20	0.300 ± 0.006	0.302 ± 0.012	0.252	0.259	0.259	0.252 ± 0.015
	1	40	0.099 ± 0.006	0.091 ± 0.009	0.081	0.069	0.069	0.081 ± 0.014
	1	30	0.068 ± 0.004	0.062 ± 0.003	0.060	0.064	0.064	0.055 ± 0.005
	1	18	0.090 ± 0.008	0.090 ± 0.012	0.071	0.052	0.052	0.068 ± 0.007
	2	20/09	0.216 ± 0.006	0.203 ± 0.009	0.172	0.207	0.200	0.164 ± 0.011
	2	10/03	0.284 ± 0.005	0.268 ± 0.007	0.231	0.215	0.189	0.253 ± 0.025
	2	20/01	0.211 ± 0.007	0.217 ± 0.008	0.160	0.168	0.191	0.165 ± 0.010
	2	42/04	0.116 ± 0.004	0.105 ± 0.012	0.072	0.067	0.067	0.071 ± 0.013
	2	40/01	0.091 ± 0.005	0.082 ± 0.006	0.066	0.056	0.047	0.061 ± 0.009
	2	01/03	0.063 ± 0.009	0.054 ± 0.005	0.058	0.053	0.048	0.056 ± 0.007
	3	11/02/03	0.268 ± 0.008	0.251 ± 0.009	0.196	0.226	0.215	0.195 ± 0.018
	3	10/03/01	0.212 ± 0.005	0.201 ± 0.010	0.153	0.159	0.156	0.178 ± 0.020
	3	43/01/03	0.172 ± 0.008	0.161 ± 0.007	0.142	0.132	0.132	0.128 ± 0.008
	3	12/01/01	0.633 ± 0.016	0.657 ± 0.026	0.326	0.462	0.480	0.288 ± 0.052
	3	16/19/03	0.058 ± 0.003	0.055 ± 0.010	0.044	0.046	0.041	0.053 ± 0.011
	3	20/09/07	0.047 ± 0.005	0.047 ± 0.008	0.041	0.038	0.044	0.055 ± 0.016
	4	11/02/03/04	0.234 ± 0.009	0.214 ± 0.006	0.173	0.206	0.215	0.168 ± 0.015
	4	43/01/03/05	0.145 ± 0.009	0.134 ± 0.008	0.113	0.092	0.111	0.089 ± 0.006
	4	11/02/03/01	0.112 ± 0.009	0.113 ± 0.006	0.072	0.074	0.084	0.078 ± 0.015
	Média		0.206	0.198	0.152	0.161	0.162	0.149
DAG	1	GO:0044464	0.969 ± 0.002	0.967 ± 0.002	0.967	0.955	0.955	0.961 ± 0.004
	1	GO:0009987	0.871 ± 0.005	0.879 ± 0.004	0.860	0.871	0.871	0.850 ± 0.012
	1	GO:0008152	0.749 ± 0.007	0.733 ± 0.006	0.726	0.720	0.720	0.710 ± 0.017
	1	GO:0005215	0.202 ± 0.008	0.193 ± 0.094	0.140	0.180	0.180	0.120 ± 0.009
	1	GO:0005198	0.517 ± 0.013	0.518 ± 0.014	0.319	0.363	0.363	0.279 ± 0.043
	1	GO:0030528	0.188 ± 0.014	0.190 ± 0.016	0.115	0.145	0.145	0.125 ± 0.020
	2	GO:0044424	0.929 ± 0.003	0.928 ± 0.002	0.898	0.897	0.897	0.898 ± 0.010
	2	GO:0044237	0.704 ± 0.009	0.689 ± 0.008	0.676	0.668	0.654	0.656 ± 0.015
	2	GO:0044238	0.674 ± 0.006	0.661 ± 0.006	0.650	0.627	0.663	0.630 ± 0.018
	2	GO:0006091	0.251 ± 0.022	0.230 ± 0.024	0.112	0.133	0.133	0.105 ± 0.010
	2	GO:0042221	0.088 ± 0.012	0.064 ± 0.005	0.075	0.067	0.074	0.064 ± 0.008
	2	GO:0045184	0.059 ± 0.003	0.070 ± 0.006	0.070	0.055	0.058	0.069 ± 0.010
	3	GO:0043229	0.606 ± 0.012	0.600 ± 0.007	0.571	0.585	0.562	0.559 ± 0.015
	3	GO:0044444	0.673 ± 0.009	0.652 ± 0.007	0.586	0.582	0.564	0.566 ± 0.022
	3	GO:0044446	0.635 ± 0.006	0.627 ± 0.011	0.580	0.568	0.546	0.550 ± 0.019
	3	GO:0006519	0.204 ± 0.035	0.151 ± 0.021	0.105	0.142	0.091	0.072 ± 0.012
	3	GO:0003677	0.167 ± 0.016	0.084 ± 0.008	0.084	0.101	0.083	0.086 ± 0.018
	3	GO:0016788	0.051 ± 0.003	0.053 ± 0.010	0.055	0.047	0.044	0.053 ± 0.017
	4	GO:0043231	0.575 ± 0.010	0.573 ± 0.009	0.524	0.549	0.493	0.505 ± 0.015
	4	GO:0044267	0.462 ± 0.016	0.443 ± 0.007	0.358	0.369	0.345	0.333 ± 0.019
	4	GO:0044428	0.458 ± 0.007	0.452 ± 0.012	0.377	0.346	0.366	0.343 ± 0.025
	4	GO:0065003	0.109 ± 0.002	0.113 ± 0.007	0.121	0.070	0.073	0.101 ± 0.008
	4	GO:0044265	0.158 ± 0.026	0.154 ± 0.026	0.090	0.066	0.072	0.079 ± 0.019
	4	GO:0043285	0.100 ± 0.011	0.113 ± 0.026	0.082	0.083	0.078	0.077 ± 0.024
	5	GO:0005634	0.307 ± 0.005	0.299 ± 0.017	0.282	0.255	0.241	0.268 ± 0.016
	5	GO:0005739	0.435 ± 0.014	0.432 ± 0.021	0.304	0.322	0.325	0.284 ± 0.024
	5	GO:0007001	0.216 ± 0.007	0.202 ± 0.008	0.203	0.140	0.147	0.164 ± 0.013
	5	GO:0006412	0.587 ± 0.032	0.578 ± 0.021	0.361	0.236	0.518	0.323 ± 0.050
	5	GO:0006396	0.273 ± 0.028	0.269 ± 0.014	0.221	0.162	0.177	0.194 ± 0.028
	5	GO:0006520	0.208 ± 0.034	0.139 ± 0.034	0.107	0.071	0.162	0.072 ± 0.014
	6	GO:0043687	0.098 ± 0.007	0.092 ± 0.007	0.117	0.094	0.096	0.112 ± 0.016
	6	GO:0045449	0.148 ± 0.010	0.140 ± 0.010	0.117	0.134	0.102	0.121 ± 0.016
	6	GO:0017111	0.060 ± 0.005	0.076 ± 0.007	0.089	0.073	0.060	0.076 ± 0.009
	6	GO:0032200	0.086 ± 0.007	0.070 ± 0.015	0.080	0.063	0.062	0.076 ± 0.013
	7	GO:0006355	0.141 ± 0.009	0.136 ± 0.013	0.119	0.137	0.091	0.116 ± 0.014
	7	GO:0000723	0.083 ± 0.008	0.069 ± 0.012	0.080	0.063	0.062	0.076 ± 0.013
	7	GO:0016887	0.049 ± 0.004	0.058 ± 0.006	0.073	0.064	0.048	0.056 ± 0.008
	Média		0.354	0.343	0.305	0.297	0.301	0.290

Tabela C.2: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Church

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.410 ± 0.004	0.407 ± 0.007	0.406	0.392	0.392	0.398 ± 0.009
	1	14	0.295 ± 0.006	0.290 ± 0.009	0.284	0.282	0.282	0.268 ± 0.011
	1	20	0.231 ± 0.004	0.230 ± 0.004	0.241	0.233	0.233	0.237 ± 0.006
	1	40	0.064 ± 0.004	0.064 ± 0.004	0.070	0.063	0.063	0.063 ± 0.002
	1	30	0.071 ± 0.007	0.058 ± 0.004	0.069	0.071	0.071	0.058 ± 0.006
	1	18	0.063 ± 0.003	0.060 ± 0.003	0.056	0.053	0.053	0.058 ± 0.002
	2	20.09	0.155 ± 0.002	0.161 ± 0.007	0.164	0.157	0.154	0.157 ± 0.006
	2	10.03	0.218 ± 0.001	0.210 ± 0.002	0.208	0.214	0.203	0.201 ± 0.003
	2	20.01	0.163 ± 0.006	0.149 ± 0.005	0.157	0.160	0.164	0.157 ± 0.003
	2	42.04	0.061 ± 0.005	0.065 ± 0.003	0.074	0.065	0.063	0.068 ± 0.005
	2	01.03	0.052 ± 0.004	0.054 ± 0.006	0.050	0.058	0.084	0.053 ± 0.010
	2	40.01	0.052 ± 0.002	0.056 ± 0.005	0.053	0.049	0.044	0.053 ± 0.002
	3	11.02.03	0.183 ± 0.003	0.184 ± 0.003	0.188	0.179	0.182	0.185 ± 0.011
	3	10.03.01	0.137 ± 0.002	0.137 ± 0.002	0.140	0.137	0.132	0.142 ± 0.002
	3	43.01.03	0.131 ± 0.005	0.124 ± 0.006	0.129	0.104	0.104	0.123 ± 0.007
	3	12.01.01	0.501 ± 0.019	0.461 ± 0.011	0.364	0.410	0.419	0.290 ± 0.012
	3	16.19.03	0.037 ± 0.001	0.044 ± 0.004	0.039	0.037	0.038	0.043 ± 0.002
	3	20.09.07	0.035 ± 0.003	0.037 ± 0.005	0.049	0.036	0.047	0.040 ± 0.003
	4	11.02.03.04	0.154 ± 0.002	0.161 ± 0.003	0.162	0.160	0.158	0.156 ± 0.009
	4	43.01.03.05	0.092 ± 0.002	0.092 ± 0.003	0.080	0.072	0.079	0.087 ± 0.006
	4	11.02.03.01	0.062 ± 0.002	0.058 ± 0.003	0.067	0.053	0.063	0.064 ± 0.004
		Média	0.151	0.148	0.145	0.142	0.144	0.138
DAG	1	GO0044464	0.957 ± 0.004	0.965 ± 0.003	0.963	0.965	0.965	0.962 ± 0.002
	1	GO0009987	0.816 ± 0.005	0.806 ± 0.014	0.821	0.803	0.803	0.818 ± 0.008
	1	GO0008152	0.741 ± 0.004	0.729 ± 0.011	0.723	0.703	0.703	0.704 ± 0.012
	1	GO0005215	0.129 ± 0.008	0.121 ± 0.009	0.110	0.127	0.122	0.129 ± 0.010
	1	GO0005198	0.372 ± 0.011	0.356 ± 0.019	0.307	0.305	0.305	0.323 ± 0.031
	1	GO0030528	0.097 ± 0.003	0.103 ± 0.006	0.105	0.110	0.110	0.106 ± 0.007
	2	GO0044424	0.884 ± 0.004	0.881 ± 0.006	0.882	0.887	0.884	0.878 ± 0.005
	2	GO0044237	0.687 ± 0.004	0.681 ± 0.009	0.662	0.636	0.661	0.645 ± 0.012
	2	GO0044238	0.647 ± 0.004	0.642 ± 0.007	0.625	0.624	0.587	0.612 ± 0.011
	2	GO0006091	0.208 ± 0.007	0.208 ± 0.027	0.109	0.149	0.155	0.142 ± 0.017
	2	GO0042221	0.063 ± 0.006	0.068 ± 0.012	0.061	0.060	0.058	0.081 ± 0.007
	2	GO0045184	0.053 ± 0.002	0.055 ± 0.005	0.060	0.065	0.062	0.062 ± 0.004
	3	GO0043229	0.530 ± 0.002	0.535 ± 0.005	0.538	0.538	0.507	0.537 ± 0.011
	3	GO0044444	0.630 ± 0.004	0.621 ± 0.005	0.588	0.598	0.571	0.595 ± 0.013
	3	GO0044446	0.505 ± 0.003	0.513 ± 0.005	0.495	0.500	0.480	0.506 ± 0.013
	3	GO0015980	0.200 ± 0.005	0.196 ± 0.024	0.105	0.155	0.139	0.128 ± 0.023
	3	GO0003677	0.079 ± 0.006	0.081 ± 0.004	0.075	0.078	0.102	0.075 ± 0.005
	3	GO0016788	0.046 ± 0.002	0.046 ± 0.004	0.049	0.046	0.052	0.047 ± 0.004
	4	GO0043231	0.498 ± 0.003	0.495 ± 0.005	0.499	0.507	0.492	0.500 ± 0.012
	4	GO0044267	0.347 ± 0.005	0.345 ± 0.008	0.316	0.320	0.338	0.327 ± 0.014
	4	GO0044428	0.278 ± 0.003	0.278 ± 0.005	0.254	0.259	0.255	0.256 ± 0.010
	4	GO0065003	0.083 ± 0.005	0.078 ± 0.009	0.081	0.087	0.082	0.088 ± 0.006
	4	GO0044265	0.113 ± 0.005	0.107 ± 0.010	0.062	0.058	0.104	0.073 ± 0.012
	4	GO0043285	0.061 ± 0.003	0.065 ± 0.008	0.052	0.061	0.066	0.054 ± 0.004
	5	GO0005634	0.231 ± 0.002	0.234 ± 0.003	0.231	0.230	0.223	0.219 ± 0.008
	5	GO0005739	0.270 ± 0.007	0.272 ± 0.009	0.250	0.232	0.249	0.247 ± 0.019
	5	GO0007001	0.162 ± 0.004	0.159 ± 0.009	0.164	0.138	0.131	0.143 ± 0.008
	5	GO0006412	0.419 ± 0.026	0.426 ± 0.029	0.320	0.314	0.356	0.332 ± 0.025
	5	GO0006396	0.071 ± 0.003	0.071 ± 0.008	0.079	0.063	0.093	0.081 ± 0.007
	5	GO0006520	0.067 ± 0.006	0.063 ± 0.010	0.077	0.041	0.055	0.060 ± 0.008
	6	GO0043687	0.117 ± 0.005	0.107 ± 0.013	0.104	0.101	0.104	0.097 ± 0.005
	6	GO0045449	0.100 ± 0.003	0.110 ± 0.007	0.106	0.094	0.098	0.101 ± 0.010
	6	GO0017111	0.057 ± 0.003	0.063 ± 0.007	0.059	0.064	0.054	0.058 ± 0.003
	6	GO0032200	0.080 ± 0.007	0.079 ± 0.012	0.097	0.073	0.053	0.081 ± 0.011
	7	GO0006355	0.096 ± 0.003	0.103 ± 0.007	0.102	0.084	0.099	0.095 ± 0.008
	7	GO0000723	0.080 ± 0.007	0.080 ± 0.018	0.097	0.073	0.053	0.081 ± 0.011
	7	GO0016887	0.042 ± 0.003	0.046 ± 0.005	0.043	0.048	0.041	0.042 ± 0.003
		Média	0.292	0.292	0.278	0.276	0.276	0.278

Tabela C.3: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Derisi

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.457 ± 0.009	0.433 ± 0.014	0.376	0.372	0.372	0.403 ± 0.008
	1	14	0.300 ± 0.003	0.301 ± 0.010	0.280	0.270	0.270	0.291 ± 0.010
	1	20	0.270 ± 0.004	0.259 ± 0.006	0.259	0.238	0.238	0.261 ± 0.009
	1	40	0.064 ± 0.002	0.065 ± 0.005	0.064	0.066	0.066	0.065 ± 0.002
	1	30	0.075 ± 0.004	0.071 ± 0.008	0.061	0.067	0.067	0.061 ± 0.005
	1	18	0.055 ± 0.002	0.059 ± 0.005	0.055	0.074	0.074	0.063 ± 0.007
	2	20.09	0.209 ± 0.006	0.193 ± 0.008	0.164	0.157	0.168	0.176 ± 0.012
	2	10.03	0.193 ± 0.003	0.195 ± 0.006	0.163	0.201	0.185	0.187 ± 0.011
	2	11.02	0.220 ± 0.004	0.212 ± 0.010	0.178	0.200	0.186	0.200 ± 0.006
	2	42.04	0.064 ± 0.003	0.066 ± 0.003	0.059	0.064	0.051	0.065 ± 0.005
	2	01.03	0.053 ± 0.003	0.056 ± 0.005	0.051	0.058	0.056	0.060 ± 0.005
	2	40.01	0.052 ± 0.003	0.053 ± 0.004	0.053	0.061	0.050	0.053 ± 0.003
	3	11.02.03	0.209 ± 0.003	0.204 ± 0.006	0.166	0.190	0.190	0.188 ± 0.006
	3	10.03.01	0.143 ± 0.004	0.145 ± 0.009	0.115	0.137	0.140	0.130 ± 0.013
	3	43.01.03	0.135 ± 0.006	0.127 ± 0.005	0.115	0.099	0.099	0.124 ± 0.006
	3	10.01.05	0.062 ± 0.004	0.068 ± 0.008	0.055	0.064	0.066	0.072 ± 0.009
	3	12.01.01	0.506 ± 0.020	0.457 ± 0.019	0.385	0.349	0.402	0.385 ± 0.017
	3	16.19.03	0.045 ± 0.006	0.052 ± 0.005	0.038	0.062	0.037	0.046 ± 0.010
	4	11.02.03.04	0.182 ± 0.004	0.182 ± 0.006	0.137	0.162	0.133	0.160 ± 0.004
	4	43.01.03.05	0.093 ± 0.006	0.095 ± 0.008	0.081	0.069	0.074	0.083 ± 0.005
	4	11.02.03.01	0.072 ± 0.003	0.069 ± 0.004	0.064	0.072	0.059	0.071 ± 0.013
		Média	0.165	0.160	0.139	0.144	0.142	0.150
DAG	1	GO0044464	0.972 ± 0.002	0.973 ± 0.002	0.965	0.971	0.971	0.965 ± 0.004
	1	GO0009987	0.854 ± 0.003	0.850 ± 0.002	0.849	0.823	0.823	0.838 ± 0.006
	1	GO0008152	0.733 ± 0.003	0.724 ± 0.005	0.736	0.672	0.672	0.687 ± 0.015
	1	GO0005215	0.168 ± 0.006	0.151 ± 0.012	0.113	0.133	0.133	0.111 ± 0.010
	1	GO0030528	0.113 ± 0.002	0.113 ± 0.005	0.119	0.098	0.098	0.108 ± 0.008
	1	GO0005198	0.401 ± 0.012	0.396 ± 0.010	0.347	0.343	0.343	0.310 ± 0.025
	2	GO0044424	0.894 ± 0.002	0.886 ± 0.007	0.888	0.874	0.865	0.878 ± 0.007
	2	GO0044237	0.677 ± 0.002	0.667 ± 0.008	0.674	0.629	0.616	0.638 ± 0.013
	2	GO0044238	0.655 ± 0.002	0.646 ± 0.007	0.643	0.600	0.634	0.619 ± 0.016
	2	GO0042221	0.066 ± 0.006	0.066 ± 0.006	0.065	0.085	0.064	0.058 ± 0.003
	2	GO0045184	0.059 ± 0.004	0.073 ± 0.007	0.067	0.058	0.058	0.072 ± 0.009
	2	GO0009719	0.061 ± 0.004	0.070 ± 0.004	0.060	0.056	0.060	0.059 ± 0.004
	3	GO0043229	0.560 ± 0.007	0.545 ± 0.004	0.525	0.499	0.497	0.518 ± 0.012
	3	GO0044444	0.590 ± 0.004	0.585 ± 0.005	0.582	0.567	0.528	0.549 ± 0.018
	3	GO0044446	0.576 ± 0.005	0.544 ± 0.013	0.533	0.512	0.500	0.515 ± 0.013
	3	GO0006519	0.077 ± 0.004	0.070 ± 0.012	0.063	0.047	0.053	0.057 ± 0.009
	3	GO0003677	0.082 ± 0.006	0.086 ± 0.007	0.081	0.063	0.073	0.083 ± 0.004
	3	GO0016788	0.053 ± 0.002	0.055 ± 0.006	0.050	0.050	0.049	0.048 ± 0.005
	4	GO0043231	0.511 ± 0.002	0.518 ± 0.007	0.501	0.476	0.499	0.478 ± 0.014
	4	GO0044267	0.390 ± 0.007	0.347 ± 0.009	0.350	0.345	0.319	0.330 ± 0.016
	4	GO0044428	0.337 ± 0.010	0.325 ± 0.018	0.316	0.307	0.268	0.305 ± 0.011
	4	GO0065003	0.091 ± 0.002	0.090 ± 0.008	0.090	0.064	0.072	0.091 ± 0.009
	4	GO0044265	0.077 ± 0.004	0.074 ± 0.008	0.096	0.058	0.067	0.081 ± 0.015
	4	GO0043285	0.052 ± 0.003	0.052 ± 0.003	0.059	0.049	0.055	0.061 ± 0.007
	5	GO0005634	0.299 ± 0.009	0.275 ± 0.012	0.253	0.231	0.244	0.242 ± 0.012
	5	GO0005739	0.292 ± 0.005	0.294 ± 0.008	0.269	0.222	0.226	0.234 ± 0.011
	5	GO0007001	0.135 ± 0.004	0.140 ± 0.012	0.165	0.155	0.141	0.141 ± 0.008
	5	GO0044451	0.095 ± 0.003	0.093 ± 0.011	0.115	0.110	0.097	0.098 ± 0.007
	5	GO0006396	0.163 ± 0.013	0.127 ± 0.017	0.125	0.068	0.105	0.129 ± 0.015
	5	GO0006412	0.458 ± 0.014	0.362 ± 0.021	0.417	0.290	0.404	0.378 ± 0.039
	6	GO0043687	0.111 ± 0.006	0.102 ± 0.014	0.108	0.090	0.103	0.101 ± 0.007
	6	GO0045449	0.110 ± 0.004	0.111 ± 0.013	0.126	0.108	0.092	0.109 ± 0.007
	6	GO0017111	0.068 ± 0.003	0.068 ± 0.005	0.065	0.079	0.060	0.072 ± 0.008
	6	GO0032200	0.063 ± 0.006	0.063 ± 0.010	0.084	0.073	0.052	0.068 ± 0.010
	7	GO0006355	0.106 ± 0.004	0.106 ± 0.011	0.124	0.103	0.097	0.103 ± 0.006
	7	GO0000723	0.063 ± 0.006	0.065 ± 0.010	0.084	0.073	0.052	0.068 ± 0.010
	7	GO0016887	0.052 ± 0.003	0.050 ± 0.006	0.048	0.067	0.037	0.050 ± 0.007
		Média	0.299	0.291	0.291	0.272	0.271	0.277

Tabela C.4: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Eisen

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.513 ± 0.010	0.514 ± 0.007	0.392	0.444	0.444	0.380 ± 0.018
	1	14	0.434 ± 0.004	0.448 ± 0.008	0.396	0.352	0.352	0.356 ± 0.030
	1	16	0.338 ± 0.009	0.339 ± 0.010	0.270	0.284	0.284	0.277 ± 0.007
	1	02	0.495 ± 0.014	0.485 ± 0.019	0.420	0.265	0.265	0.239 ± 0.040
	1	40	0.100 ± 0.010	0.102 ± 0.011	0.074	0.063	0.063	0.065 ± 0.012
	1	18	0.077 ± 0.005	0.083 ± 0.008	0.08	0.053	0.053	0.075 ± 0.018
	2	20.09	0.262 ± 0.010	0.266 ± 0.013	0.215	0.217	0.222	0.192 ± 0.020
	2	11.02	0.341 ± 0.016	0.307 ± 0.014	0.248	0.247	0.266	0.222 ± 0.023
	2	10.03	0.249 ± 0.017	0.248 ± 0.014	0.244	0.229	0.267	0.198 ± 0.016
	2	18.02	0.069 ± 0.003	0.076 ± 0.005	0.080	0.053	0.054	0.074 ± 0.018
	2	40.01	0.088 ± 0.011	0.090 ± 0.012	0.059	0.052	0.057	0.051 ± 0.006
	2	42.04	0.122 ± 0.009	0.119 ± 0.013	0.078	0.063	0.077	0.075 ± 0.008
	3	11.02.03	0.345 ± 0.020	0.291 ± 0.016	0.236	0.244	0.215	0.212 ± 0.027
	3	10.03.01	0.191 ± 0.010	0.195 ± 0.017	0.181	0.169	0.154	0.152 ± 0.014
	3	43.01.03	0.158 ± 0.004	0.152 ± 0.018	0.117	0.113	0.113	0.111 ± 0.012
	3	14.13.01	0.317 ± 0.014	0.298 ± 0.041	0.177	0.123	0.159	0.153 ± 0.041
	3	20.09.07	0.073 ± 0.013	0.076 ± 0.028	0.072	0.064	0.067	0.056 ± 0.017
	3	20.09.18	0.091 ± 0.012	0.068 ± 0.007	0.042	0.054	0.075	0.041 ± 0.005
	4	11.02.03.04	0.319 ± 0.029	0.267 ± 0.014	0.208	0.203	0.230	0.172 ± 0.024
	4	11.02.03.01	0.130 ± 0.009	0.129 ± 0.007	0.091	0.085	0.130	0.092 ± 0.012
	4	43.01.03.05	0.146 ± 0.010	0.135 ± 0.018	0.090	0.088	0.102	0.089 ± 0.011
		Média	0.231	0.223	0.180	0.165	0.174	0.156
DAG	1	GO0044464	0.980 ± 0.003	0.982 ± 0.002	0.981	0.975	0.975	0.977 ± 0.004
	1	GO0009987	0.899 ± 0.003	0.902 ± 0.007	0.916	0.877	0.877	0.907 ± 0.005
	1	GO0008152	0.827 ± 0.006	0.825 ± 0.012	0.808	0.807	0.807	0.792 ± 0.016
	1	GO0005215	0.239 ± 0.016	0.225 ± 0.021	0.174	0.154	0.154	0.149 ± 0.024
	1	GO0030528	0.225 ± 0.006	0.222 ± 0.014	0.137	0.212	0.212	0.139 ± 0.017
	1	GO0005198	0.641 ± 0.008	0.638 ± 0.013	0.537	0.536	0.536	0.503 ± 0.041
	2	GO0044424	0.948 ± 0.002	0.946 ± 0.005	0.926	0.917	0.928	0.922 ± 0.006
	2	GO0044237	0.762 ± 0.009	0.765 ± 0.011	0.751	0.742	0.695	0.736 ± 0.010
	2	GO0044238	0.754 ± 0.007	0.747 ± 0.007	0.716	0.722	0.696	0.710 ± 0.013
	2	GO0045184	0.075 ± 0.006	0.074 ± 0.008	0.121	0.071	0.066	0.080 ± 0.012
	2	GO0042221	0.092 ± 0.010	0.097 ± 0.025	0.077	0.118	0.144	0.070 ± 0.012
	2	GO0043228	0.245 ± 0.012	0.265 ± 0.024	0.192	0.076	0.110	0.138 ± 0.040
	3	GO0044444	0.738 ± 0.009	0.739 ± 0.009	0.651	0.607	0.609	0.636 ± 0.022
	3	GO0043229	0.646 ± 0.010	0.626 ± 0.005	0.585	0.608	0.611	0.581 ± 0.013
	3	GO0044446	0.710 ± 0.008	0.698 ± 0.008	0.640	0.663	0.648	0.620 ± 0.019
	3	GO0003723	0.093 ± 0.006	0.116 ± 0.014	0.090	0.073	0.062	0.080 ± 0.013
	3	GO0016788	0.055 ± 0.004	0.060 ± 0.008	0.051	0.055	0.045	0.047 ± 0.005
	3	GO0030529	0.140 ± 0.008	0.174 ± 0.022	0.116	0.101	0.116	0.109 ± 0.017
	4	GO0043231	0.617 ± 0.011	0.608 ± 0.006	0.557	0.588	0.585	0.549 ± 0.016
	4	GO0044267	0.564 ± 0.009	0.547 ± 0.007	0.472	0.449	0.380	0.476 ± 0.025
	4	GO0044428	0.491 ± 0.006	0.493 ± 0.011	0.331	0.302	0.350	0.329 ± 0.026
	4	GO0048523	0.094 ± 0.007	0.097 ± 0.014	0.092	0.079	0.064	0.090 ± 0.016
	4	GO0006886	0.080 ± 0.007	0.072 ± 0.007	0.129	0.104	0.070	0.075 ± 0.009
	4	GO0009892	0.083 ± 0.006	0.088 ± 0.014	0.097	0.049	0.052	0.086 ± 0.019
	5	GO0005634	0.312 ± 0.014	0.333 ± 0.017	0.287	0.261	0.239	0.242 ± 0.013
	5	GO0005739	0.490 ± 0.012	0.522 ± 0.011	0.394	0.462	0.385	0.360 ± 0.028
	5	GO0007001	0.278 ± 0.007	0.277 ± 0.020	0.207	0.197	0.178	0.197 ± 0.032
	5	GO0006396	0.277 ± 0.009	0.284 ± 0.023	0.128	0.169	0.159	0.158 ± 0.040
	5	GO0006520	0.329 ± 0.018	0.239 ± 0.044	0.079	0.055	0.099	0.092 ± 0.016
	5	GO0006323	0.173 ± 0.008	0.142 ± 0.021	0.109	0.067	0.124	0.120 ± 0.024
	6	GO0045449	0.181 ± 0.010	0.170 ± 0.017	0.204	0.150	0.118	0.143 ± 0.007
	6	GO0017111	0.078 ± 0.004	0.109 ± 0.021	0.127	0.081	0.085	0.086 ± 0.006
	6	GO0043687	0.104 ± 0.005	0.098 ± 0.006	0.106	0.083	0.087	0.105 ± 0.012
	6	GO0032200	0.092 ± 0.006	0.085 ± 0.014	0.080	0.090	0.069	0.080 ± 0.013
	6	GO0006325	0.166 ± 0.008	0.125 ± 0.027	0.109	0.072	0.124	0.120 ± 0.024
	7	GO0006355	0.164 ± 0.009	0.164 ± 0.014	0.176	0.140	0.109	0.136 ± 0.006
	7	GO0000723	0.087 ± 0.005	0.086 ± 0.015	0.080	0.090	0.069	0.080 ± 0.013
	7	GO0016887	0.075 ± 0.005	0.103 ± 0.028	0.123	0.080	0.100	0.069 ± 0.004
		Média	0.363	0.362	0.325	0.313	0.309	0.310

Tabela C.5: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Gaschl

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.530 ± 0.007	0.528 ± 0.007	0.444	0.443	0.443	0.434 ± 0.009
	1	14	0.437 ± 0.007	0.448 ± 0.006	0.329	0.319	0.319	0.340 ± 0.012
	1	20	0.323 ± 0.006	0.317 ± 0.009	0.300	0.260	0.260	0.270 ± 0.009
	1	40	0.089 ± 0.006	0.094 ± 0.010	0.067	0.066	0.066	0.065 ± 0.006
	1	30	0.073 ± 0.006	0.088 ± 0.014	0.054	0.057	0.057	0.069 ± 0.014
	1	18	0.062 ± 0.003	0.062 ± 0.004	0.065	0.046	0.046	0.065 ± 0.009
	2	20.09	0.214 ± 0.006	0.214 ± 0.010	0.184	0.163	0.176	0.191 ± 0.015
	2	10.03	0.340 ± 0.007	0.331 ± 0.010	0.213	0.217	0.252	0.222 ± 0.025
	2	20.01	0.264 ± 0.007	0.243 ± 0.014	0.245	0.186	0.155	0.190 ± 0.008
	2	42.04	0.117 ± 0.007	0.102 ± 0.008	0.071	0.065	0.066	0.073 ± 0.007
	2	01.03	0.195 ± 0.012	0.176 ± 0.018	0.070	0.082	0.088	0.058 ± 0.008
	2	40.01	0.077 ± 0.008	0.088 ± 0.010	0.056	0.058	0.092	0.054 ± 0.004
	3	11.02.03	0.290 ± 0.005	0.308 ± 0.010	0.212	0.209	0.214	0.218 ± 0.010
	3	10.03.01	0.221 ± 0.006	0.220 ± 0.007	0.146	0.155	0.188	0.161 ± 0.023
	3	43.01.03	0.169 ± 0.003	0.174 ± 0.013	0.128	0.119	0.119	0.120 ± 0.010
	3	12.01.01	0.842 ± 0.011	0.687 ± 0.046	0.660	0.664	0.523	0.623 ± 0.055
	3	16.19.03	0.084 ± 0.006	0.086 ± 0.007	0.049	0.046	0.045	0.057 ± 0.013
	3	20.09.07	0.076 ± 0.006	0.084 ± 0.008	0.052	0.059	0.047	0.067 ± 0.016
	4	11.02.03.04	0.249 ± 0.006	0.255 ± 0.008	0.172	0.169	0.162	0.173 ± 0.007
	4	43.01.03.05	0.131 ± 0.006	0.136 ± 0.009	0.095	0.087	0.085	0.085 ± 0.010
	4	11.02.03.01	0.174 ± 0.004	0.182 ± 0.014	0.088	0.095	0.098	0.097 ± 0.013
		Média	0.236	0.230	0.176	0.170	0.167	0.173
DAG	1	GO0044464	0.974 ± 0.002	0.970 ± 0.003	0.963	0.964	0.964	0.959 ± 0.002
	1	GO0009987	0.873 ± 0.008	0.874 ± 0.006	0.855	0.857	0.857	0.856 ± 0.003
	1	GO0008152	0.785 ± 0.005	0.777 ± 0.005	0.733	0.724	0.724	0.712 ± 0.008
	1	GO0005215	0.285 ± 0.016	0.252 ± 0.020	0.194	0.189	0.189	0.121 ± 0.009
	1	GO0005198	0.642 ± 0.027	0.638 ± 0.021	0.495	0.494	0.494	0.434 ± 0.028
	1	GO0030528	0.174 ± 0.006	0.186 ± 0.015	0.142	0.158	0.158	0.133 ± 0.019
	2	GO0044424	0.945 ± 0.001	0.941 ± 0.003	0.912	0.905	0.888	0.896 ± 0.010
	2	GO0044237	0.740 ± 0.007	0.733 ± 0.005	0.674	0.695	0.677	0.673 ± 0.010
	2	GO0044238	0.712 ± 0.007	0.711 ± 0.005	0.660	0.640	0.647	0.653 ± 0.010
	2	GO0042221	0.156 ± 0.019	0.110 ± 0.024	0.074	0.083	0.107	0.070 ± 0.013
	2	GO0045184	0.095 ± 0.008	0.101 ± 0.007	0.066	0.093	0.069	0.074 ± 0.011
	2	GO0003735	0.862 ± 0.042	0.872 ± 0.073	0.659	0.698	0.730	0.593 ± 0.034
	3	GO0043229	0.605 ± 0.006	0.609 ± 0.006	0.611	0.578	0.578	0.552 ± 0.007
	3	GO0044444	0.699 ± 0.007	0.686 ± 0.012	0.647	0.618	0.578	0.578 ± 0.011
	3	GO0044446	0.693 ± 0.013	0.670 ± 0.014	0.617	0.620	0.596	0.587 ± 0.015
	3	GO0015980	0.227 ± 0.023	0.221 ± 0.028	0.224	0.153	0.153	0.108 ± 0.016
	3	GO0003677	0.101 ± 0.015	0.122 ± 0.014	0.075	0.074	0.077	0.082 ± 0.013
	3	GO0016788	0.063 ± 0.015	0.053 ± 0.005	0.049	0.073	0.053	0.051 ± 0.010
	4	GO0043231	0.580 ± 0.009	0.587 ± 0.009	0.576	0.540	0.527	0.486 ± 0.010
	4	GO0044267	0.581 ± 0.008	0.560 ± 0.016	0.398	0.456	0.453	0.417 ± 0.017
	4	GO0044428	0.510 ± 0.012	0.497 ± 0.014	0.409	0.404	0.377	0.395 ± 0.024
	4	GO0065003	0.129 ± 0.008	0.119 ± 0.013	0.116	0.069	0.100	0.116 ± 0.007
	4	GO0044265	0.227 ± 0.025	0.217 ± 0.025	0.116	0.129	0.111	0.107 ± 0.023
	4	GO0043285	0.177 ± 0.023	0.186 ± 0.025	0.087	0.124	0.112	0.098 ± 0.024
	5	GO0005634	0.339 ± 0.015	0.327 ± 0.016	0.265	0.270	0.267	0.258 ± 0.014
	5	GO0005739	0.488 ± 0.006	0.467 ± 0.019	0.430	0.371	0.292	0.250 ± 0.011
	5	GO0007001	0.240 ± 0.012	0.226 ± 0.018	0.218	0.171	0.127	0.182 ± 0.019
	5	GO0006412	0.785 ± 0.042	0.645 ± 0.064	0.583	0.514	0.640	0.539 ± 0.029
	5	GO0006396	0.280 ± 0.016	0.192 ± 0.063	0.205	0.139	0.124	0.238 ± 0.028
	5	GO0006520	0.472 ± 0.034	0.324 ± 0.072	0.085	0.108	0.281	0.108 ± 0.043
	6	GO0043687	0.129 ± 0.008	0.139 ± 0.015	0.108	0.106	0.106	0.099 ± 0.004
	6	GO0045449	0.161 ± 0.009	0.167 ± 0.016	0.136	0.093	0.122	0.126 ± 0.008
	6	GO0017111	0.059 ± 0.005	0.083 ± 0.007	0.082	0.088	0.063	0.071 ± 0.006
	6	GO0032200	0.085 ± 0.008	0.078 ± 0.011	0.104	0.068	0.050	0.089 ± 0.010
	7	GO0006355	0.142 ± 0.010	0.155 ± 0.013	0.131	0.090	0.084	0.122 ± 0.007
	7	GO0000723	0.083 ± 0.008	0.080 ± 0.010	0.104	0.068	0.050	0.089 ± 0.010
	7	GO0016887	0.046 ± 0.004	0.062 ± 0.006	0.060	0.075	0.051	0.050 ± 0.004
		Média	0.409	0.396	0.348	0.338	0.337	0.324

Tabela C.6: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Gasch2

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.485 ± 0.012	0.452 ± 0.013	0.451	0.449	0.449	0.422 ± 0.021
	1	14	0.380 ± 0.009	0.374 ± 0.008	0.342	0.294	0.294	0.297 ± 0.026
	1	20	0.285 ± 0.008	0.273 ± 0.007	0.253	0.272	0.272	0.260 ± 0.017
	1	40	0.108 ± 0.012	0.122 ± 0.011	0.076	0.060	0.060	0.065 ± 0.008
	1	30	0.066 ± 0.004	0.075 ± 0.011	0.060	0.053	0.053	0.061 ± 0.008
	1	18	0.078 ± 0.005	0.089 ± 0.007	0.070	0.054	0.054	0.070 ± 0.012
	2	20.09	0.186 ± 0.008	0.189 ± 0.006	0.179	0.173	0.183	0.172 ± 0.012
	2	10.03	0.283 ± 0.006	0.254 ± 0.008	0.199	0.189	0.182	0.196 ± 0.015
	2	20.01	0.214 ± 0.006	0.197 ± 0.008	0.205	0.182	0.169	0.185 ± 0.018
	2	42.04	0.091 ± 0.008	0.083 ± 0.006	0.068	0.068	0.057	0.065 ± 0.005
	2	01.03	0.072 ± 0.008	0.060 ± 0.010	0.060	0.056	0.056	0.054 ± 0.006
	2	40.01	0.094 ± 0.005	0.101 ± 0.008	0.062	0.047	0.049	0.054 ± 0.007
	3	11.02.03	0.290 ± 0.007	0.283 ± 0.009	0.226	0.193	0.219	0.221 ± 0.023
	3	10.03.01	0.197 ± 0.007	0.173 ± 0.008	0.145	0.135	0.195	0.142 ± 0.011
	3	43.01.03	0.143 ± 0.004	0.147 ± 0.005	0.132	0.111	0.111	0.112 ± 0.006
	3	12.01.01	0.685 ± 0.006	0.693 ± 0.012	0.627	0.621	0.612	0.481 ± 0.047
	3	16.19.03	0.076 ± 0.004	0.074 ± 0.008	0.059	0.075	0.049	0.049 ± 0.008
	3	20.09.07	0.045 ± 0.004	0.055 ± 0.007	0.069	0.056	0.055	0.052 ± 0.008
	4	11.02.03.04	0.244 ± 0.008	0.232 ± 0.008	0.186	0.167	0.154	0.176 ± 0.018
	4	43.01.03.05	0.122 ± 0.004	0.124 ± 0.008	0.088	0.104	0.089	0.086 ± 0.010
	4	11.02.03.01	0.117 ± 0.004	0.121 ± 0.007	0.084	0.082	0.092	0.090 ± 0.014
	Média		0.203	0.199	0.173	0.164	0.165	0.158
DAG	1	GO0044464	0.961 ± 0.003	0.964 ± 0.004	0.966	0.965	0.965	0.961 ± 0.002
	1	GO0009987	0.856 ± 0.006	0.865 ± 0.005	0.863	0.837	0.837	0.842 ± 0.004
	1	GO0008152	0.735 ± 0.003	0.729 ± 0.005	0.741	0.739	0.739	0.716 ± 0.007
	1	GO0005215	0.173 ± 0.004	0.168 ± 0.016	0.181	0.157	0.157	0.127 ± 0.009
	1	GO0005198	0.529 ± 0.003	0.540 ± 0.009	0.460	0.469	0.469	0.424 ± 0.033
	1	GO0030528	0.175 ± 0.011	0.179 ± 0.011	0.106	0.122	0.122	0.113 ± 0.008
	2	GO0044424	0.925 ± 0.005	0.926 ± 0.003	0.910	0.905	0.895	0.901 ± 0.004
	2	GO0044237	0.694 ± 0.004	0.685 ± 0.006	0.696	0.686	0.674	0.663 ± 0.009
	2	GO0044238	0.677 ± 0.007	0.669 ± 0.004	0.667	0.637	0.655	0.643 ± 0.009
	2	GO0042221	0.099 ± 0.015	0.080 ± 0.011	0.076	0.095	0.077	0.069 ± 0.004
	2	GO0045184	0.081 ± 0.007	0.079 ± 0.006	0.062	0.060	0.057	0.068 ± 0.006
	2	GO0003735	0.704 ± 0.018	0.715 ± 0.014	0.622	0.640	0.660	0.553 ± 0.051
	3	GO0043229	0.633 ± 0.003	0.619 ± 0.007	0.589	0.587	0.600	0.567 ± 0.008
	3	GO0044444	0.676 ± 0.006	0.650 ± 0.005	0.590	0.551	0.590	0.565 ± 0.009
	3	GO0044446	0.665 ± 0.003	0.652 ± 0.009	0.601	0.602	0.590	0.581 ± 0.017
	3	GO0006519	0.303 ± 0.021	0.275 ± 0.020	0.175	0.195	0.156	0.126 ± 0.015
	3	GO0003677	0.099 ± 0.023	0.105 ± 0.014	0.072	0.086	0.080	0.082 ± 0.010
	3	GO0016788	0.044 ± 0.002	0.046 ± 0.004	0.043	0.050	0.051	0.045 ± 0.004
	4	GO0043231	0.598 ± 0.006	0.589 ± 0.007	0.533	0.522	0.542	0.515 ± 0.009
	4	GO0044267	0.483 ± 0.014	0.465 ± 0.008	0.436	0.439	0.407	0.417 ± 0.021
	4	GO0044428	0.472 ± 0.007	0.461 ± 0.009	0.392	0.356	0.376	0.355 ± 0.020
	4	GO0065003	0.102 ± 0.005	0.114 ± 0.008	0.103	0.084	0.099	0.102 ± 0.006
	4	GO0044265	0.153 ± 0.014	0.163 ± 0.013	0.122	0.099	0.109	0.108 ± 0.013
	4	GO0043285	0.128 ± 0.018	0.130 ± 0.017	0.098	0.084	0.092	0.100 ± 0.013
	5	GO0005634	0.303 ± 0.002	0.292 ± 0.007	0.287	0.239	0.234	0.253 ± 0.009
	5	GO0005739	0.450 ± 0.006	0.457 ± 0.007	0.282	0.304	0.307	0.264 ± 0.013
	5	GO0007001	0.246 ± 0.007	0.212 ± 0.026	0.156	0.199	0.139	0.165 ± 0.011
	5	GO0044451	0.159 ± 0.007	0.154 ± 0.014	0.129	0.114	0.097	0.120 ± 0.013
	5	GO0006412	0.572 ± 0.006	0.578 ± 0.015	0.521	0.484	0.593	0.490 ± 0.042
	5	GO0006396	0.239 ± 0.012	0.256 ± 0.021	0.217	0.111	0.141	0.193 ± 0.021
	6	GO0043687	0.111 ± 0.008	0.100 ± 0.008	0.109	0.086	0.100	0.115 ± 0.009
	6	GO0045449	0.149 ± 0.018	0.138 ± 0.012	0.116	0.102	0.093	0.113 ± 0.006
	6	GO0017111	0.069 ± 0.006	0.070 ± 0.004	0.074	0.101	0.070	0.075 ± 0.004
	6	GO0032200	0.073 ± 0.005	0.073 ± 0.008	0.063	0.081	0.057	0.072 ± 0.005
	7	GO0006355	0.139 ± 0.016	0.131 ± 0.012	0.105	0.097	0.101	0.109 ± 0.005
	7	GO0000723	0.071 ± 0.004	0.075 ± 0.009	0.063	0.081	0.057	0.072 ± 0.005
	7	GO0016887	0.057 ± 0.007	0.052 ± 0.003	0.057	0.103	0.052	0.058 ± 0.004
	Média		0.368	0.364	0.332	0.326	0.325	0.317

Tabela C.7: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Pheno

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.401 ± 0.004	0.386 ± 0.013	0.401	0.384	0.384	0.419 ± 0.010
	1	10	0.245 ± 0.008	0.233 ± 0.010	0.245	0.215	0.215	0.219 ± 0.005
	1	14	0.297 ± 0.013	0.291 ± 0.008	0.281	0.278	0.278	0.284 ± 0.017
	1	40	0.077 ± 0.004	0.075 ± 0.006	0.081	0.070	0.070	0.073 ± 0.006
	1	30	0.056 ± 0.004	0.072 ± 0.010	0.050	0.058	0.058	0.065 ± 0.019
	1	18	0.087 ± 0.015	0.087 ± 0.012	0.076	0.057	0.057	0.074 ± 0.014
	2	10.03	0.176 ± 0.010	0.165 ± 0.011	0.178	0.163	0.199	0.158 ± 0.011
	2	20.09	0.171 ± 0.007	0.185 ± 0.014	0.180	0.155	0.160	0.163 ± 0.003
	2	01.05	0.156 ± 0.009	0.150 ± 0.011	0.159	0.166	0.161	0.175 ± 0.007
	2	20.03	0.051 ± 0.006	0.054 ± 0.006	0.088	0.039	0.044	0.050 ± 0.010
	2	14.04	0.087 ± 0.008	0.086 ± 0.012	0.100	0.089	0.075	0.068 ± 0.010
	2	18.02	0.083 ± 0.014	0.094 ± 0.010	0.076	0.057	0.057	0.074 ± 0.014
	3	11.02.03	0.174 ± 0.012	0.172 ± 0.011	0.178	0.158	0.151	0.163 ± 0.006
	3	10.03.01	0.144 ± 0.015	0.122 ± 0.013	0.147	0.137	0.143	0.116 ± 0.012
	3	43.01.03	0.120 ± 0.006	0.123 ± 0.005	0.137	0.139	0.139	0.128 ± 0.011
	3	12.01.01	0.048 ± 0.004	0.055 ± 0.010	0.047	0.075	0.108	0.049 ± 0.002
	3	34.11.03	0.063 ± 0.004	0.065 ± 0.010	0.062	0.064	0.087	0.064 ± 0.006
	3	16.19.03	0.039 ± 0.003	0.036 ± 0.001	0.041	0.037	0.037	0.036 ± 0.003
	4	11.02.03.04	0.148 ± 0.013	0.145 ± 0.010	0.145	0.137	0.136	0.138 ± 0.005
	4	43.01.03.05	0.093 ± 0.013	0.099 ± 0.010	0.098	0.114	0.101	0.107 ± 0.012
Média			0.136	0.135	0.139	0.130	0.133	0.131
DAG	1	GO0044464	0.953 ± 0.008	0.963 ± 0.005	0.970	0.963	0.963	0.959 ± 0.002
	1	GO0009987	0.869 ± 0.004	0.862 ± 0.007	0.863	0.852	0.852	0.844 ± 0.006
	1	GO0008152	0.640 ± 0.007	0.631 ± 0.008	0.647	0.641	0.641	0.655 ± 0.008
	1	GO0005215	0.072 ± 0.004	0.078 ± 0.004	0.077	0.083	0.083	0.083 ± 0.004
	1	GO0030528	0.081 ± 0.006	0.086 ± 0.009	0.091	0.076	0.076	0.082 ± 0.003
	1	GO0005198	0.098 ± 0.011	0.100 ± 0.007	0.091	0.117	0.117	0.102 ± 0.004
	2	GO0044424	0.887 ± 0.007	0.879 ± 0.006	0.881	0.886	0.882	0.872 ± 0.006
	2	GO0044237	0.585 ± 0.004	0.582 ± 0.009	0.584	0.611	0.585	0.600 ± 0.007
	2	GO0044238	0.565 ± 0.004	0.556 ± 0.009	0.562	0.584	0.595	0.582 ± 0.010
	2	GO0043228	0.069 ± 0.006	0.071 ± 0.006	0.082	0.067	0.064	0.090 ± 0.004
	2	GO0042221	0.072 ± 0.007	0.099 ± 0.013	0.072	0.119	0.095	0.073 ± 0.006
	2	GO0007154	0.058 ± 0.011	0.058 ± 0.008	0.059	0.082	0.086	0.063 ± 0.013
	3	GO0043229	0.528 ± 0.010	0.512 ± 0.017	0.531	0.521	0.540	0.530 ± 0.014
	3	GO0044444	0.470 ± 0.008	0.456 ± 0.011	0.514	0.493	0.493	0.524 ± 0.007
	3	GO0044446	0.504 ± 0.006	0.488 ± 0.011	0.491	0.504	0.493	0.475 ± 0.008
	3	GO0003677	0.058 ± 0.004	0.061 ± 0.004	0.070	0.066	0.074	0.066 ± 0.005
	3	GO0005975	0.059 ± 0.003	0.065 ± 0.005	0.073	0.074	0.063	0.068 ± 0.003
	3	GO0048519	0.054 ± 0.007	0.052 ± 0.004	0.090	0.053	0.049	0.088 ± 0.006
	4	GO0043231	0.497 ± 0.010	0.493 ± 0.020	0.500	0.511	0.513	0.494 ± 0.012
	4	GO0044267	0.218 ± 0.014	0.218 ± 0.023	0.239	0.260	0.234	0.281 ± 0.019
	4	GO0044428	0.226 ± 0.022	0.214 ± 0.024	0.270	0.289	0.259	0.241 ± 0.013
	4	GO0043285	0.057 ± 0.012	0.060 ± 0.010	0.052	0.046	0.053	0.049 ± 0.003
	4	GO0065003	0.059 ± 0.006	0.061 ± 0.014	0.091	0.056	0.068	0.098 ± 0.005
	4	GO0048523	0.051 ± 0.006	0.052 ± 0.006	0.082	0.053	0.047	0.081 ± 0.007
	5	GO0005634	0.214 ± 0.016	0.220 ± 0.016	0.217	0.207	0.252	0.210 ± 0.007
	5	GO0005739	0.247 ± 0.008	0.216 ± 0.019	0.289	0.272	0.259	0.280 ± 0.004
	5	GO0007001	0.125 ± 0.012	0.130 ± 0.013	0.155	0.168	0.144	0.143 ± 0.005
	5	GO0006412	0.059 ± 0.002	0.064 ± 0.009	0.106	0.057	0.078	0.100 ± 0.009
	5	GO0044451	0.077 ± 0.011	0.096 ± 0.011	0.096	0.106	0.086	0.112 ± 0.014
	5	GO0006396	0.057 ± 0.007	0.067 ± 0.011	0.063	0.059	0.054	0.066 ± 0.004
	6	GO0043687	0.086 ± 0.004	0.088 ± 0.008	0.090	0.090	0.097	0.093 ± 0.012
	6	GO0045449	0.083 ± 0.007	0.085 ± 0.004	0.094	0.093	0.089	0.094 ± 0.003
	6	GO0017111	0.067 ± 0.005	0.088 ± 0.011	0.093	0.070	0.067	0.088 ± 0.002
	6	GO0032200	0.052 ± 0.005	0.056 ± 0.010	0.061	0.070	0.079	0.064 ± 0.006
	7	GO0006355	0.084 ± 0.008	0.083 ± 0.003	0.093	0.093	0.069	0.094 ± 0.003
	7	GO0000723	0.050 ± 0.003	0.057 ± 0.009	0.061	0.070	0.079	0.064 ± 0.006
	7	GO0016887	0.053 ± 0.005	0.072 ± 0.011	0.088	0.059	0.055	0.081 ± 0.003
Média			0.243	0.244	0.256	0.255	0.252	0.256

Tabela C.8: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Spo

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.476 ± 0.006	0.440 ± 0.009	0.389	0.405	0.405	0.402 ± 0.008
	1	14	0.388 ± 0.005	0.368 ± 0.006	0.338	0.321	0.321	0.333 ± 0.012
	1	20	0.263 ± 0.005	0.280 ± 0.011	0.243	0.231	0.231	0.245 ± 0.009
	1	40	0.070 ± 0.002	0.068 ± 0.005	0.069	0.072	0.072	0.077 ± 0.007
	1	30	0.057 ± 0.002	0.066 ± 0.008	0.066	0.054	0.054	0.070 ± 0.008
	1	18	0.073 ± 0.006	0.067 ± 0.006	0.068	0.052	0.052	0.067 ± 0.005
	2	20.09	0.200 ± 0.003	0.193 ± 0.009	0.166	0.162	0.211	0.181 ± 0.009
	2	10.03	0.277 ± 0.008	0.260 ± 0.020	0.261	0.255	0.272	0.260 ± 0.014
	2	11.02	0.186 ± 0.005	0.177 ± 0.011	0.184	0.176	0.169	0.191 ± 0.008
	2	01.03	0.047 ± 0.003	0.046 ± 0.004	0.059	0.067	0.057	0.052 ± 0.004
	2	42.04	0.069 ± 0.003	0.070 ± 0.005	0.082	0.069	0.063	0.093 ± 0.009
	2	40.01	0.059 ± 0.006	0.057 ± 0.003	0.057	0.063	0.060	0.069 ± 0.009
	3	11.02.03	0.172 ± 0.007	0.159 ± 0.007	0.179	0.177	0.157	0.184 ± 0.009
	3	10.03.01	0.152 ± 0.003	0.140 ± 0.009	0.157	0.147	0.139	0.159 ± 0.010
	3	43.01.03	0.216 ± 0.005	0.197 ± 0.011	0.126	0.158	0.158	0.165 ± 0.031
	3	10.01.05	0.124 ± 0.010	0.104 ± 0.012	0.121	0.103	0.095	0.116 ± 0.010
	3	12.01.01	0.533 ± 0.028	0.520 ± 0.028	0.499	0.570	0.549	0.418 ± 0.018
	3	16.19.03	0.056 ± 0.007	0.048 ± 0.004	0.042	0.041	0.037	0.047 ± 0.009
	4	11.02.03.04	0.142 ± 0.005	0.133 ± 0.006	0.154	0.151	0.162	0.158 ± 0.011
	4	43.01.03.05	0.089 ± 0.003	0.083 ± 0.004	0.086	0.085	0.082	0.102 ± 0.013
	4	11.02.03.01	0.071 ± 0.004	0.062 ± 0.004	0.063	0.064	0.060	0.068 ± 0.002
	Média		0.177	0.168	0.162	0.163	0.162	0.165
DAG	1	GO0044464	0.953 ± 0.003	0.955 ± 0.004	0.953	0.965	0.965	0.955 ± 0.002
	1	GO0009987	0.826 ± 0.002	0.827 ± 0.006	0.823	0.801	0.801	0.823 ± 0.015
	1	GO0008152	0.709 ± 0.005	0.702 ± 0.006	0.708	0.699	0.699	0.699 ± 0.019
	1	GO0005215	0.110 ± 0.008	0.113 ± 0.007	0.112	0.109	0.109	0.109 ± 0.005
	1	GO0030528	0.103 ± 0.010	0.109 ± 0.009	0.098	0.094	0.094	0.092 ± 0.007
	1	GO0005198	0.413 ± 0.006	0.410 ± 0.022	0.362	0.365	0.365	0.355 ± 0.025
	2	GO0044424	0.890 ± 0.003	0.882 ± 0.004	0.881	0.877	0.868	0.875 ± 0.009
	2	GO0044237	0.652 ± 0.004	0.648 ± 0.010	0.649	0.618	0.636	0.648 ± 0.019
	2	GO0044238	0.620 ± 0.004	0.613 ± 0.007	0.624	0.601	0.615	0.623 ± 0.024
	2	GO0042221	0.066 ± 0.007	0.067 ± 0.006	0.081	0.072	0.068	0.065 ± 0.008
	2	GO0045184	0.071 ± 0.006	0.079 ± 0.004	0.079	0.063	0.092	0.074 ± 0.005
	2	GO0009719	0.071 ± 0.002	0.066 ± 0.007	0.070	0.070	0.066	0.071 ± 0.015
	3	GO0043229	0.540 ± 0.005	0.544 ± 0.005	0.535	0.516	0.522	0.539 ± 0.009
	3	GO0044444	0.556 ± 0.010	0.526 ± 0.011	0.544	0.529	0.512	0.537 ± 0.009
	3	GO0044446	0.542 ± 0.010	0.542 ± 0.009	0.511	0.512	0.516	0.520 ± 0.010
	3	GO0006519	0.163 ± 0.018	0.094 ± 0.022	0.109	0.061	0.111	0.128 ± 0.026
	3	GO0003677	0.070 ± 0.007	0.073 ± 0.007	0.091	0.073	0.100	0.088 ± 0.018
	3	GO0016788	0.052 ± 0.003	0.052 ± 0.008	0.042	0.062	0.046	0.052 ± 0.006
	4	GO0043231	0.492 ± 0.006	0.496 ± 0.008	0.489	0.477	0.484	0.493 ± 0.011
	4	GO0044267	0.361 ± 0.014	0.338 ± 0.007	0.372	0.358	0.318	0.369 ± 0.027
	4	GO0044428	0.322 ± 0.004	0.311 ± 0.011	0.250	0.261	0.267	0.286 ± 0.017
	4	GO0065003	0.092 ± 0.004	0.092 ± 0.009	0.112	0.075	0.084	0.089 ± 0.019
	4	GO0044265	0.155 ± 0.009	0.132 ± 0.025	0.125	0.138	0.150	0.101 ± 0.020
	4	GO0043285	0.088 ± 0.006	0.072 ± 0.007	0.080	0.144	0.110	0.093 ± 0.025
	5	GO0005634	0.234 ± 0.010	0.224 ± 0.011	0.240	0.210	0.218	0.224 ± 0.015
	5	GO0005739	0.268 ± 0.007	0.252 ± 0.008	0.252	0.237	0.231	0.254 ± 0.014
	5	GO0007001	0.160 ± 0.006	0.143 ± 0.010	0.160	0.167	0.136	0.149 ± 0.011
	5	GO0006412	0.481 ± 0.026	0.465 ± 0.019	0.427	0.310	0.439	0.428 ± 0.055
	5	GO0006396	0.221 ± 0.007	0.190 ± 0.022	0.096	0.101	0.136	0.169 ± 0.025
	5	GO0006520	0.164 ± 0.015	0.065 ± 0.012	0.103	0.044	0.143	0.126 ± 0.026
	6	GO0043687	0.129 ± 0.012	0.096 ± 0.011	0.117	0.093	0.086	0.107 ± 0.009
	6	GO0045449	0.108 ± 0.006	0.098 ± 0.012	0.092	0.089	0.082	0.093 ± 0.009
	6	GO0017111	0.082 ± 0.011	0.081 ± 0.010	0.078	0.069	0.057	0.074 ± 0.011
	6	GO0032200	0.054 ± 0.003	0.049 ± 0.003	0.073	0.069	0.066	0.061 ± 0.005
	7	GO0006355	0.103 ± 0.006	0.095 ± 0.010	0.082	0.088	0.080	0.090 ± 0.009
	7	GO0000723	0.055 ± 0.004	0.051 ± 0.003	0.073	0.069	0.066	0.061 ± 0.005
	7	GO0016887	0.073 ± 0.016	0.064 ± 0.008	0.068	0.052	0.042	0.057 ± 0.009
	Média		0.299	0.287	0.285	0.274	0.281	0.286

Tabela C.9: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Expr

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.553 ± 0.010	0.535 ± 0.007	0.444	0.458	0.458	0.431 ± 0.018
	1	14	0.452 ± 0.009	0.457 ± 0.006	0.349	0.343	0.343	0.352 ± 0.016
	1	20	0.339 ± 0.010	0.335 ± 0.010	0.303	0.286	0.286	0.282 ± 0.012
	1	40	0.104 ± 0.011	0.091 ± 0.008	0.071	0.057	0.057	0.070 ± 0.008
	1	30	0.083 ± 0.011	0.077 ± 0.012	0.065	0.045	0.045	0.083 ± 0.016
	1	18	0.079 ± 0.008	0.076 ± 0.007	0.063	0.058	0.058	0.066 ± 0.008
	2	20.09	0.244 ± 0.011	0.231 ± 0.009	0.191	0.198	0.179	0.200 ± 0.013
	2	10.03	0.386 ± 0.015	0.368 ± 0.014	0.282	0.248	0.277	0.262 ± 0.018
	2	20.01	0.302 ± 0.012	0.281 ± 0.016	0.240	0.197	0.173	0.185 ± 0.008
	2	42.04	0.131 ± 0.012	0.128 ± 0.014	0.078	0.092	0.053	0.085 ± 0.011
	2	01.03	0.216 ± 0.016	0.136 ± 0.026	0.059	0.081	0.113	0.061 ± 0.012
	2	40.01	0.095 ± 0.011	0.082 ± 0.021	0.060	0.053	0.074	0.058 ± 0.007
	3	11.02.03	0.318 ± 0.006	0.322 ± 0.013	0.200	0.209	0.225	0.218 ± 0.019
	3	10.03.01	0.237 ± 0.005	0.227 ± 0.017	0.182	0.162	0.153	0.165 ± 0.013
	3	43.01.03	0.266 ± 0.010	0.242 ± 0.016	0.125	0.166	0.166	0.139 ± 0.019
	3	12.01.01	0.820 ± 0.025	0.716 ± 0.075	0.607	0.677	0.609	0.541 ± 0.066
	3	16.19.03	0.108 ± 0.015	0.103 ± 0.015	0.063	0.040	0.040	0.059 ± 0.010
	3	20.09.07	0.089 ± 0.022	0.101 ± 0.022	0.052	0.058	0.065	0.068 ± 0.017
	4	11.02.03.04	0.273 ± 0.009	0.273 ± 0.012	0.169	0.158	0.169	0.184 ± 0.018
	4	43.01.03.05	0.149 ± 0.013	0.146 ± 0.012	0.090	0.110	0.083	0.096 ± 0.012
	4	11.02.03.01	0.162 ± 0.008	0.163 ± 0.014	0.080	0.083	0.100	0.098 ± 0.010
	Média		0.257	0.242	0.180	0.180	0.177	0.176
DAG	1	GO0044464	0.972 ± 0.002	0.972 ± 0.003	0.970	0.969	0.969	0.964 ± 0.004
	1	GO0009987	0.867 ± 0.007	0.856 ± 0.008	0.884	0.873	0.873	0.856 ± 0.016
	1	GO0008152	0.759 ± 0.007	0.729 ± 0.013	0.759	0.702	0.702	0.728 ± 0.019
	1	GO0005215	0.339 ± 0.016	0.278 ± 0.026	0.197	0.139	0.139	0.156 ± 0.015
	1	GO0005198	0.613 ± 0.042	0.639 ± 0.041	0.478	0.481	0.481	0.411 ± 0.059
	1	GO0030528	0.209 ± 0.014	0.218 ± 0.011	0.131	0.151	0.151	0.107 ± 0.006
	2	GO0044424	0.940 ± 0.003	0.939 ± 0.005	0.915	0.908	0.910	0.897 ± 0.007
	2	GO0044237	0.704 ± 0.008	0.687 ± 0.017	0.708	0.653	0.661	0.667 ± 0.017
	2	GO0044238	0.682 ± 0.010	0.673 ± 0.016	0.672	0.606	0.655	0.642 ± 0.016
	2	GO0042221	0.122 ± 0.024	0.112 ± 0.022	0.053	0.107	0.103	0.079 ± 0.016
	2	GO0045184	0.078 ± 0.015	0.084 ± 0.006	0.069	0.091	0.052	0.063 ± 0.005
	2	GO0003735	0.851 ± 0.060	0.827 ± 0.091	0.621	0.657	0.621	0.504 ± 0.084
	3	GO0043229	0.623 ± 0.009	0.632 ± 0.010	0.633	0.568	0.557	0.548 ± 0.010
	3	GO0044444	0.703 ± 0.012	0.696 ± 0.009	0.631	0.590	0.608	0.552 ± 0.018
	3	GO0044446	0.690 ± 0.007	0.684 ± 0.012	0.607	0.617	0.623	0.574 ± 0.015
	3	GO0006519	0.463 ± 0.023	0.365 ± 0.043	0.140	0.169	0.214	0.065 ± 0.008
	3	GO0003677	0.133 ± 0.038	0.114 ± 0.011	0.089	0.083	0.094	0.074 ± 0.010
	3	GO0016788	0.052 ± 0.005	0.053 ± 0.005	0.067	0.049	0.054	0.047 ± 0.006
	4	GO0043231	0.590 ± 0.008	0.602 ± 0.008	0.594	0.537	0.503	0.506 ± 0.013
	4	GO0044267	0.535 ± 0.014	0.530 ± 0.008	0.420	0.442	0.415	0.381 ± 0.031
	4	GO0044428	0.494 ± 0.022	0.495 ± 0.021	0.377	0.396	0.390	0.300 ± 0.021
	4	GO0065003	0.117 ± 0.009	0.107 ± 0.015	0.101	0.079	0.088	0.109 ± 0.009
	4	GO0044265	0.245 ± 0.029	0.246 ± 0.017	0.110	0.096	0.141	0.077 ± 0.009
	4	GO0043285	0.197 ± 0.024	0.185 ± 0.025	0.094	0.093	0.113	0.061 ± 0.012
	5	GO0005634	0.324 ± 0.024	0.311 ± 0.015	0.280	0.250	0.258	0.232 ± 0.013
	5	GO0005739	0.529 ± 0.011	0.522 ± 0.010	0.405	0.370	0.330	0.266 ± 0.017
	5	GO0007001	0.261 ± 0.011	0.229 ± 0.022	0.197	0.160	0.190	0.153 ± 0.010
	5	GO0044451	0.197 ± 0.013	0.197 ± 0.032	0.135	0.104	0.125	0.103 ± 0.004
	5	GO0006412	0.770 ± 0.062	0.722 ± 0.029	0.576	0.457	0.595	0.459 ± 0.072
	5	GO0006396	0.304 ± 0.021	0.248 ± 0.044	0.200	0.121	0.167	0.138 ± 0.024
	6	GO0043687	0.133 ± 0.009	0.130 ± 0.015	0.097	0.087	0.101	0.099 ± 0.008
	6	GO0045449	0.172 ± 0.017	0.172 ± 0.017	0.157	0.110	0.129	0.105 ± 0.006
	6	GO0017111	0.066 ± 0.007	0.084 ± 0.011	0.087	0.096	0.065	0.066 ± 0.006
	6	GO0032200	0.096 ± 0.018	0.072 ± 0.006	0.089	0.066	0.067	0.079 ± 0.007
	7	GO0006355	0.167 ± 0.014	0.164 ± 0.014	0.144	0.104	0.089	0.100 ± 0.006
	7	GO0000723	0.092 ± 0.013	0.076 ± 0.009	0.089	0.066	0.067	0.079 ± 0.007
	7	GO0016887	0.057 ± 0.008	0.072 ± 0.012	0.056	0.086	0.071	0.050 ± 0.009
	Média		0.409	0.398	0.347	0.328	0.334	0.305

Tabela C.10: Valores de $AU(\overline{PRC})$ em classes específicas do conjunto de dados Seq

Taxonomia	Nível	Classes	Bp-Predicted	Rp-Predicted	Clus-HMC	Clus-HSC	Clus-SC	hmAnt-Miner
Árvore	1	01	0.597 ± 0.008	0.619 ± 0.006	0.502	0.463	0.463	0.445 ± 0.020
	1	14	0.362 ± 0.008	0.366 ± 0.010	0.288	0.304	0.304	0.289 ± 0.016
	1	20	0.555 ± 0.009	0.553 ± 0.014	0.496	0.346	0.346	0.364 ± 0.017
	1	40	0.111 ± 0.005	0.124 ± 0.017	0.083	0.089	0.089	0.084 ± 0.010
	1	18	0.095 ± 0.011	0.115 ± 0.018	0.065	0.063	0.063	0.076 ± 0.010
	1	30	0.122 ± 0.023	0.119 ± 0.009	0.056	0.049	0.049	0.074 ± 0.015
	2	20.09	0.351 ± 0.009	0.348 ± 0.011	0.249	0.207	0.215	0.215 ± 0.016
	2	10.03	0.311 ± 0.008	0.310 ± 0.012	0.213	0.166	0.200	0.225 ± 0.013
	2	20.01	0.508 ± 0.007	0.502 ± 0.010	0.451	0.339	0.339	0.299 ± 0.014
	2	42.04	0.121 ± 0.025	0.129 ± 0.013	0.076	0.055	0.079	0.079 ± 0.008
	2	01.03	0.076 ± 0.021	0.094 ± 0.015	0.075	0.070	0.050	0.070 ± 0.007
	2	40.01	0.106 ± 0.007	0.111 ± 0.014	0.069	0.083	0.056	0.071 ± 0.007
	3	11.02.03	0.402 ± 0.013	0.390 ± 0.020	0.262	0.229	0.236	0.269 ± 0.021
	3	10.03.01	0.202 ± 0.012	0.217 ± 0.015	0.162	0.132	0.158	0.167 ± 0.009
	3	43.01.03	0.183 ± 0.006	0.179 ± 0.011	0.141	0.134	0.134	0.138 ± 0.016
	3	10.01.05	0.089 ± 0.009	0.143 ± 0.015	0.063	0.082	0.067	0.068 ± 0.009
	3	12.01.01	0.771 ± 0.009	0.760 ± 0.018	0.504	0.426	0.480	0.502 ± 0.054
	3	34.11.03	0.125 ± 0.021	0.150 ± 0.021	0.062	0.060	0.055	0.074 ± 0.010
	4	11.02.03.04	0.386 ± 0.016	0.363 ± 0.014	0.222	0.194	0.186	0.238 ± 0.021
	4	43.01.03.05	0.162 ± 0.008	0.153 ± 0.009	0.108	0.096	0.124	0.108 ± 0.019
	4	11.02.03.01	0.133 ± 0.010	0.132 ± 0.014	0.095	0.075	0.076	0.092 ± 0.011
	Média		0.275	0.280	0.202	0.174	0.179	0.188
DAG	1	GO0044464	0.963 ± 0.007	0.964 ± 0.004	0.960	0.951	0.951	0.953 ± 0.005
	1	GO0009987	0.870 ± 0.003	0.874 ± 0.007	0.872	0.844	0.844	0.870 ± 0.007
	1	GO0008152	0.793 ± 0.011	0.793 ± 0.008	0.774	0.700	0.700	0.730 ± 0.010
	1	GO0005215	0.639 ± 0.009	0.626 ± 0.019	0.464	0.420	0.420	0.313 ± 0.015
	1	GO0005198	0.572 ± 0.010	0.586 ± 0.015	0.430	0.367	0.367	0.394 ± 0.023
	1	GO0030528	0.355 ± 0.017	0.253 ± 0.028	0.199	0.204	0.204	0.223 ± 0.027
	2	GO0044424	0.937 ± 0.004	0.933 ± 0.005	0.922	0.897	0.894	0.916 ± 0.005
	2	GO0044237	0.740 ± 0.013	0.740 ± 0.006	0.714	0.694	0.686	0.685 ± 0.012
	2	GO0044238	0.698 ± 0.012	0.702 ± 0.009	0.664	0.632	0.634	0.651 ± 0.014
	2	GO0006091	0.173 ± 0.020	0.182 ± 0.024	0.165	0.085	0.084	0.107 ± 0.019
	2	GO0042221	0.065 ± 0.010	0.081 ± 0.011	0.065	0.054	0.059	0.073 ± 0.008
	2	GO0045184	0.131 ± 0.012	0.154 ± 0.015	0.064	0.087	0.081	0.071 ± 0.017
	3	GO0043229	0.594 ± 0.009	0.602 ± 0.007	0.584	0.554	0.547	0.561 ± 0.011
	3	GO0044444	0.648 ± 0.019	0.638 ± 0.009	0.629	0.546	0.530	0.570 ± 0.008
	3	GO0044446	0.686 ± 0.007	0.679 ± 0.009	0.649	0.610	0.564	0.615 ± 0.016
	3	GO0015980	0.172 ± 0.026	0.186 ± 0.022	0.155	0.080	0.093	0.103 ± 0.021
	3	GO0006519	0.188 ± 0.033	0.211 ± 0.031	0.088	0.185	0.092	0.072 ± 0.009
	3	GO0003677	0.178 ± 0.022	0.166 ± 0.015	0.106	0.089	0.111	0.142 ± 0.025
	4	GO0043231	0.556 ± 0.010	0.567 ± 0.005	0.535	0.500	0.485	0.514 ± 0.008
	4	GO0044267	0.455 ± 0.007	0.454 ± 0.008	0.383	0.337	0.329	0.376 ± 0.024
	4	GO0044428	0.495 ± 0.015	0.492 ± 0.015	0.446	0.353	0.346	0.409 ± 0.026
	4	GO0065003	0.137 ± 0.013	0.153 ± 0.021	0.116	0.094	0.092	0.106 ± 0.014
	4	GO0044265	0.083 ± 0.013	0.087 ± 0.008	0.107	0.055	0.074	0.076 ± 0.009
	4	GO0043285	0.065 ± 0.006	0.064 ± 0.005	0.065	0.052	0.055	0.064 ± 0.006
	5	GO0005634	0.314 ± 0.038	0.316 ± 0.012	0.327	0.241	0.283	0.322 ± 0.019
	5	GO0005739	0.389 ± 0.018	0.386 ± 0.016	0.308	0.284	0.310	0.244 ± 0.010
	5	GO0007001	0.203 ± 0.008	0.205 ± 0.012	0.170	0.149	0.155	0.160 ± 0.007
	5	GO0006412	0.682 ± 0.013	0.681 ± 0.014	0.491	0.440	0.501	0.435 ± 0.028
	5	GO0044451	0.159 ± 0.010	0.143 ± 0.014	0.124	0.082	0.083	0.108 ± 0.007
	5	GO0006396	0.146 ± 0.020	0.129 ± 0.017	0.102	0.071	0.089	0.089 ± 0.010
	6	GO0043687	0.220 ± 0.030	0.208 ± 0.016	0.105	0.096	0.120	0.115 ± 0.014
	6	GO0045449	0.224 ± 0.017	0.198 ± 0.009	0.167	0.106	0.128	0.188 ± 0.027
	6	GO0017111	0.159 ± 0.055	0.270 ± 0.037	0.134	0.190	0.200	0.071 ± 0.005
	6	GO0032200	0.085 ± 0.006	0.083 ± 0.011	0.085	0.064	0.056	0.082 ± 0.010
	7	GO0006355	0.220 ± 0.018	0.192 ± 0.008	0.173	0.100	0.117	0.174 ± 0.022
	7	GO0000723	0.084 ± 0.006	0.085 ± 0.009	0.085	0.064	0.056	0.082 ± 0.010
	7	GO0016887	0.115 ± 0.045	0.231 ± 0.037	0.135	0.158	0.112	0.049 ± 0.003
	Média		0.384	0.387	0.339	0.309	0.309	0.317