HELYANE BRONOSKI BORGES

CLASSIFICADOR HIERÁRQUICO MULTIRRÓTULO USANDO UMA REDE NEURAL COMPETITIVA

Tese de Doutorado apresentado ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná, como requisito parcial para obtenção do título de Doutor em Informática.

CURITIBA

2012

HELYANE BRONOSKI BORGES

CLASSIFICADOR HIERÁRQUICO MULTIRRÓTULO USANDO UMA REDE NEURAL COMPETITIVA

Tese de Doutorado apresentado ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná, como requisito parcial para obtenção do título de Doutor em Informática.

Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. Júlio Cesar Nievola

CURITIBA

2012

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central

Borges, Helyane Bronoski
B732c Classificador hierárquico multirrótulo usando uma rede neural competitiva /
2012 Helyane Bronoski Borges ; orientador, Júlio César Nievola. -- 2012.
165 f. : il. ; 30 cm

Tese (doutorado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2012
Bibliografia: f. 132-140

1. Redes neurais (computação). 2. Computação evolucionária. 3. Biologia computacional. 4. Informática. I. Nievola, Júlio César. II. Pontifícia Universidade
do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD 20. ed. – 004

Pontifícia Universidade Católica do Paraná
Escola Politécnica
Programa de Pós-Graduação em Informática

ATA DE DEFESA DE TESE DE DOUTORADO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

ÁREA DE CONCENTRAÇÃO: CIÊNCIA DA COMPUTAÇÃO

DEFESA DE TESE DE DOUTORADO Nº 014/2012

Aos 05 dias de Outubro de 2012 realizou-se a sessão pública de Defesa da Tese de Doutorado intitulada **"Classificador Hierárquico Multirrótulo usando uma Rede Neural Competitiva"** apresentada pela aluna **Helyane Bronoski Borges** como requisito parcial para a obtenção do título de Doutor em Informática, perante uma Banca Examinadora composta pelos seguintes membros:

Prof. Dr. Julio Cesar Nievola PUCPR (Orientador)	Julio Cesar Nievola *(assinatura)*	APROVADO *(aprov/reprov.)*
Prof. Dr. Emerson Cabrera Paraiso PUCPR		Aprov
Prof. Dr. Edson Emilio Scalabrin PUCPR		Aprov
Prof. Dr. Roberto Tadeu Raittz UFPR		Aprovado
Prof. Dr. Carlos Nascimento Silla Junior UTFPR	Carlos N. Silla Jr.	APROV.

Conforme as normas regimentais do PPGIa e da PUCPR, o trabalho apresentado foi considerado APROVADO (*aprovado/reprovado*), segundo avaliação da maioria dos membros desta Banca Examinadora. Este resultado está condicionado ao cumprimento integral das solicitações da Banca Examinadora registradas no Livro de Defesas do programa.

Prof. Dr. Fabrício Enembreck
Diretor do Programa de Pós-Graduação em Informática

Programa de Pós-Graduação em Informática PPGIa PUCPR

Rua Imaculada Conceição, 1155 Prado Velho CEP 80215 901 Curitiba Paraná Brasil
Tel.: (41) 3271 1669 Fax.: (41) 3271 2121 www.ppgia.pucpr.br

Dedicatória

Aos meus pais pela dedicação e constante apoio.

AGRADECIMENTOS

Primeiramente agradeço a Deus que meu deu saúde e sabedoria para vencer todos os obstáculos pelos quais passei durante a realização deste trabalho.

Ao meu pai João D. Borges que não pode estar pessoalmente compartilhando este momento comigo, mas que no tempo que passamos juntos sempre me incentivou e me apoiou nas decisões. Sei que, ao lado de Deus, ele está feliz e orgulhoso comigo.

A minha mãe Dilma B. Borges, pela força que me deu durante esses anos de estudo, principalmente na hora mais difícil da minha vida, em que o mundo parecia desabar em minha cabeça.

Ao meu orientador, Júlio Cesar Nievola, pela confiança, incentivo, paciência e pelos ensinamentos passados ao longo do desenvolvimento deste trabalho. Espero poder continuar pesquisando com esse grande Mestre.

A minha irmã Gilyane B. Borges que acompanhou todo o sacrifício durante esses quatro anos de pesquisa.

Ao meu namorado Fábio Junior Alves Batista por ter entrado na minha vida em um momento difícil e estar ao meu lado sempre.

A minha querida amiga Simone Nasser Matos que tinha sempre uma palavra amiga nos momentos difíceis e de desânimo.

A minha amiga Regiane A. Siqueira que nunca negou esforços em me auxiliar nas frequentes dúvidas em matemática e estatística.

Aos colegas de pesquisa Ricardo Cerri e Mauri Ferrandin pela auxilio e pela troca de conhecimentos.

Aos colegas de trabalho do departamento de informática da UTFPR, Câmpus Ponta Grossa, em especial ao Prof. Flávio Vieira.

E a todos aqueles que de alguma maneira contribuíram para que esse trabalho fosse realizado.

PUBLICAÇÕES

BORGES, Helyane Bronoski; NIEVOLA, J. C. Hierarchical Classification using a Competitive Neural Network. In: 8th International Conference on Natural Computation (ICNC'12), 2012, Chongqing, China. 8th International Conference on Natural Computation (ICNC'12). Piscataway, NJ: IEEE Press, 2012. v. 1. p. 172-177.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Hierarchical Classification Using a Competitive Neural Network for Protein Function Prediction. In: 14th International Conference on Artificial Intelligence (ICAI'12), 2012, Las Vegas. 14th International Conference on Artificial Intelligence (ICAI'12). USA: CSREA Press, 2012. v. 1. p. 1-7.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Multi-Label Hierarchical Classification using a Competitive Neural Network for Protein Function Prediction. In: 2012 International Joint Conference on Neural Networks (IJCNN 2012), 2012, Brisbane, Austrália. 2012 International Joint Conference on Neural Networks (IJCNN 2012). Piscataway, NJ: IEEE Press, 2012. v. 1. p. 1-8.

BORGES, Helyane Bronoski; NIEVOLA, Julio Cesar. Classificador Hierárquico Multi-Classe usando uma Rede Neural Competitiva. In: XIX SEMIC. Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação, 2011, Curitiba. Caderno de Resumos do XIX Seminário de Iniciação Científica. XIII Mostra de Pesquisa da Pós-Graduação. Curitiba: Champagnat, 2011. v. 1. p. 137-137.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Classificador Hierárquico Multiclasses usando uma Rede Neural Competitiva para Predição Funcional de Proteínas. In: XVIII SEMIC. Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação, 2010, Curitiba. Caderno de Resumos do XVIII Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação. Curitiba: Champagnat, 2010. v. 1. p. 281-281.

RESUMO

Esta tese propõe um novo algoritmo baseado em Redes Neurais Artificiais (RNA) Competitivas para classificação hierárquica multirrótulo usando a abordagem de classificação global, no qual o classificador processa e avalia todas as classes da hierárquica em uma única vez. Esse tipo de abordagem, juntamente com a predição multirrótulo aplicada em estruturas hierárquicas do tipo grafo acíclico dirigido é considerada complexa e desafiadora, visto que um exemplo está associado a uma ou mais classes da hierarquia de classes. Além disso, uma segunda versão do classificador é desenvolvida no qual é modificada a forma de treinamento da rede neural que será por meio da técnica de estratégia evolucionária. Dados biológicos de 10 bases de dados da Ontologia Gênica são usados para a realização de experimentos computacionais. Os critérios utilizados para a avaliação dos algoritmos são a correção preditiva através da medida de distância dependente de profundidade e a medida hierárquica baseada na ancestralidade. Para poderem ser comparados com outros classificadores da literatura, tal qual o Clus-HMC e Clus-HSC, os classificadores são adaptados para serem avaliados por meio da curva de precisão e revocação. Os resultados mostram que quando comparado às duas propostas desenvolvidas, a primeira apresenta resultados superiores. Já quando comparado com o Clus-HMC e Clus-HSC às abordagens propostas obtiveram resultados equivalentes.

Palavras-chave: Classificação Hierárquica Multirrótulo, Abordagem de Classificação Global, Rede Neural Artificial Competitiva, Estratégia Evolucionária.

ABSTRACT

This thesis proposes a new algorithm based on Competitive Artificial Neural Networks for multi-label hierarchical classification using the global approach, in which the classifier processes and evaluates all classes in the hierarchy once. This approach, together with the multi-label prediction applied hierarchical directed acyclic graph is complex and challenging, since an example is associated with one or more classes of the class hierarchy. Also, the second version of the classifier is developed which is modified in the form of training the neural network will be using the evolutionary strategy approach. Biological data from ten databases of Gene Ontology is used to conduct computational experiments. The criteria used to assess the predictive correction algorithms were the distance-dependent depth and measures based on hierarchical ancestry. To be comparable with other classifiers from the literature, like the Clus-HMC and Clus-HSC, classifiers are adapted to be evaluated by the curve precision recall. The results showed that compared to two proposals developed, the first displays superior results. Already compared to Clus-HMC and Clus-HSC obtained similar results.

Keywords: Hierarchical Classification Multi-label, Global Approach, Competitive Artificial Neural Network, Evolutionary Strategical.

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizagem de Máquina
ART	*Adaptive Resonance Theory*
AUC	*Area Under the ROC Curve*
AUCPR	*Area Under the ROC Curve Precision Recall*
CSSA	*Condensing Sort and Select Algorithm*
CSSAG	*Condensing Sort and Select Algorithm for GraphsHMC-LMLP*
DAG	Directed Acyclic Graph
E	Especificidade
EC	Enzyme Commission
EE	Estratégia Evolucionária
FP	Falso Positivo
FN	Falso Negativo
GO	*Gene Ontology*
HIER-BFS	*Breadth-First Search Graphs*
HIER-MB	*Markov Blanket Graphs*
HMC	*Hierarchical multi-label classification*
HSC	*Hierarchical single-label classification*
LVQ	*Learning Vector Quantization*
k-NN	*k-Nearest Neighbor*
MHCAIS	*Multi-Label Hierarchical Classification with an Artificial Immune System*
MHC-CNN	*Multi-label Hierarchical Classification using a Competitive Neural Network*
MHC-ES	*Multi-label Hierarchical Classification usinga Strategical Evolutionary*
P	Precisão
PCT	*Predictive Clustering Trees*
RGOW	*Random GO Walk*
RNA	Redes Neurais Artificiais
ROC	*Receiver Operating Characteristics*
S	Sensibilidade
SC	*Single-label Classification*
SOM	*Self-organizing Map*
SVM	*Support Vector Machines*

TA	Taxa de Acerto
TDIDT	*Top-down induction of decision tree*s
TE	Taxa de Erro
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

LISTA DE EQUAÇÕES

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os algoritmos de aprendizagem de máquina (AM) podem ser classificados em duas categorias: aprendizado supervisionado e não supervisionado. A diferença entre ambos diz respeito à forma de como é feito o processo de aprendizagem. No aprendizado não supervisionado não existe conhecimento sobre o domínio. Já no aprendizado supervisionado, uma das tarefas mais utilizada é a classificação. O objetivo dessa tarefa é a construção de um modelo que descreve um conjunto pré-determinado de classes de dados. Essa construção é feita analisando-se as amostras de uma base de dados, que são descritas por atributos e cada uma delas pertence a uma classe pré-definida, identificada por um dos atributos, chamado atributo rótulo da classe ou, simplesmente, classe.

Há dois tipos de classificação de dados denominados de classificação plana e classificação hierárquica. A maioria dos trabalhos citados na literatura envolve o tipo de classificação plana na qual uma instância do conjunto de treinamento está relacionada a uma determinada classe. Há casos ainda em que deseja-se determinar uma classe para cada instância de um total de N classes em que $N>2$. Dessa maneira, a classificação acaba sendo denominada de classificação multiclasses. Além disso, existem casos em que uma instância pode estar associada a mais de uma classe. Este tipo de classificação é chamada de classificação multirrótulo.

Na classificação hierárquica as classes estão dispostas em uma hierarquia. Nesse tipo de classificação ao associar uma classe da hierarquia para uma instância automaticamente estará associando todas as suas classes ancestrais. Além disso, a classificação hierárquica também podem ser multirrótulo. Entre alguns exemplos, em que se utiliza a classificação hierárquica pode-se citar a predição de proteínas em dados de bioinformática, de textos, de gêneros musicais entre outros.

Existem algumas técnicas para tratar problemas hierárquicos. Dentre elas, podem-se citar os trabalhos dos seguintes autores, entre outros: Clare & King (2001), Blockeel (2002), Jensen et. al (2002), Clare & King (2003), Holden & Freitas (2005), Holden & Freitas (2006), para estruturas hierárquicas do tipo árvore e Laegreid et. al (2003), King et. al (2003), Tu (2004), Barutcuoglu et. al (2006), Vens et. al (2008) para estruturas do tipo DAG, o qual será o foco dessa tese. O desenvolvimento de classificadores hierárquicos pode fazer uso de duas abordagens principais: Classificador Local e Classificador Global. A principal diferença entre essas abordagens é a maneira de como o modelo de classificação é criado. O primeiro avalia

cada nó do conjunto de dados hierárquico independentemente. Já o segundo leva em consideração toda a hierarquia de classes do conjunto ao predizer uma instância.

A grande maioria dos trabalhos de classificação hierárquica para estruturas do tipo grafo acíclico dirigido, (Directed Acyclic Graph – DAG) utiliza a abordagem de classificação local. Isso ocorre porque classificadores dessa abordagem são mais simples de ser construídos e podem fazer uso de algoritmos usados na classificação plana. Já o desenvolvimento de classificadores globais é mais complexo, pois a estrutura hierárquica deve ser respeitada. Sendo assim, algoritmos de classificação plana não podem ser utilizados sem que haja uma mudança na sua estrutura.

Nesta tese são propostos dois algoritmos para a classificação hierárquica multirrótulo para estruturas do tipo DAG, utilizando a abordagem de classificador global. O primeiro algoritmo proposto utiliza uma Rede Neural Artificial (RNA) Competitiva treinado com o uso de um algoritmo semelhante ao proposto na rede *Self-organizing Map* (SOM). Já o segundo utilizada a computação evolutiva por meio da técnica de Estratégia Evolucionária (EE) para realizar o treinamento da mesma rede neural.

1.1 MOTIVAÇÃO E CONTEXTO

A classificação hierárquica vem sendo bastante utilizada em diversas áreas, como na categorização de textos, predição de proteínas, classificação de gêneros musicais e imagens entre outras.

A categorização de texto é um processo de atribuir automaticamente uma ou mais categorias a documentos textuais (SUM & LIN, 2001). Com o surgimento de bibliotecas digitais e a grande diversidade de textos que esses repositórios possuem, pesquisa nessa área tem sido motivada. Entre os principais objetivos está em melhorar o processo de recuperação de informações, pois a classificação manual ou a utilização de métodos computacionais simples torna-se inviável.

Na classificação hierárquica, cada nó da hierarquia é representado por uma classe. Um exemplo típico de uma hierarquia de classe estruturada em uma árvore, em problemas de categorização de textos é apresentado na Figura 1.

A classificação de imagens, como por exemplo, imagens médicas, e a classificação de gêneros musicais também são áreas que se utilizada a classificação hierárquica.

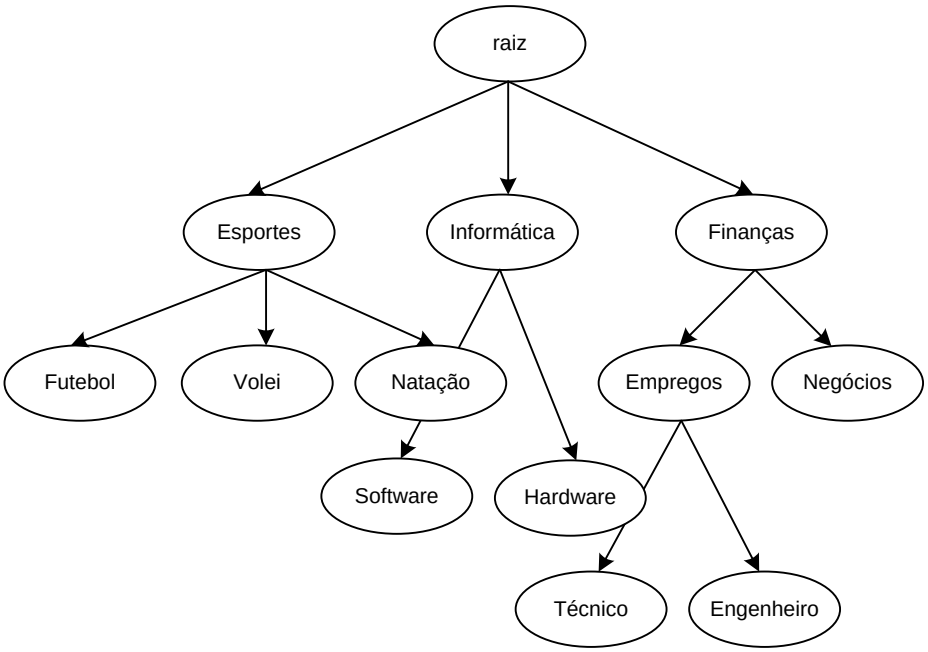


Figura 1 – Exemplo de Hierarquia para Classificação de Textos.

Outra área que motiva a pesquisa com classificação hierárquica é a bioinformática. Um dos principais objetivos dessa área é a predição da função que uma proteína exerce em uma célula (BALDI & POLLASTRI, 2002). Realizar experiências para determinar como as proteínas funcionam não é uma tarefa fácil, isso porque proteínas de estruturas semelhantes, e até mesmo de sequências semelhantes, podem ser classificadas como exercendo diferentes funções. Além disso, assim como muitas sequências diferentes são compatíveis com uma mesma estrutura, proteínas de diferentes formas podem ter a mesma função (LESK, 2008). É neste contexto que a informática com a aplicação das tarefas de mineração de dados, como é o caso da classificação hierárquica, pode auxiliar no processo de classificação funcional de proteínas.

Um exemplo típico de uma hierarquia de classe estruturada, como uma árvore, é a classificação funcional de enzimas. Cada enzima é nomeada como uma classe de acordo com seu código EC (Enzyme Commission). Os códigos EC são um esquema de classificação numérica para as enzimas, baseado nas reações químicas que catalisam. Cada número EC está associado a um nome recomendado para a referida enzima. O código de EC consiste de uma série de quatro dígitos onde o primeiro dígito especifica a classe mais geral de enzimas e o quarto dígito o mais específico. Por exemplo, o código EC 1.1.1.1 especifica a classe Álcool Dehydrogenase. Já um exemplo de classificação funcional de proteína estruturada como um DAG é a Gene Ontology (GO). A GO consiste de três categorias de funções denominadas de:

processo biológico, função molecular e componente celular que são implementadas em três ontologias independentes (GENE ONTOLOGY, 1998).

Nessa área, uma grande motivação da utilização da classificação hierárquica é o diagnóstico e tratamento de doenças e o desenvolvimento de medicamentos através do conhecimento descoberto.

Como se podem observar vários são os problemas onde a classificação hierárquica é usada. Porém, a maioria dos algoritmos para esse tipo de classificação foi desenvolvida para a estrutura do tipo árvore. Esse tipo de estrutura é mais simples, pois um nó filho tem apenas um nó pai, comparado com a estrutura DAG, em que um nó pode ter múltiplos pais. Dessa maneira, o desenvolvimento de algoritmos para essa estrutura torna-se mais complexo, pois a hierarquia de classes deve ser preservada. Com isso os resultados obtidos podem ser mais condizentes.

Outra motivação diz respeito a classificação multirrótulo. Esse tipo de problema ainda é pouco explorado e consequentemente classificadores que sejam capazes de predizer várias classes para uma mesma instância são poucos na literatura. Sendo assim, pesquisas nessa área são importantes, pois vários são os problemas que precisam ser tratados dessa maneira.

1.2 OBJETIVO DA PESQUISA

A maioria dos algoritmos para tratar problemas de classificação hierárquica foram desenvolvidos para a estrutura do tipo árvore. Os algoritmos desenvolvidos para suportar estruturas do tipo DAG, normalmente, não avaliam o modelo hierárquico como um todo (abordagem classificador global ou *big-bang*). Na abordagem de classificação global apenas um classificador é construído para distinguir entre todas as classes da hierarquia e por isso o seu desenvolvimento torna-se mais complexo em estruturas DAG. Além disso, problemas hierárquicos também podem ser multirrótulos o que acaba se tornando ainda mais difícil a construção de um modelo para a predição dos dados.

Desta forma, o objetivo principal deste trabalho é o desenvolvimento de um algoritmo para a classificação hierárquica multirrótulo, para problemas estruturados em DAG, utilizando a abordagem de classificação global. Para isso será utilizado uma rede neural artificial competitiva. A utilização de redes neurais é motivada pela sua capacidade em trabalhar com dados ruidosos e a sua alta adaptabilidade, que fornecem muitos dos requisitos necessários na área de bioinformática.

1.3 ORIGINALIDADE

A maioria dos trabalhos da literatura de classificação hierárquica são desenvolvidos para tratar de problemas estruturados em árvore. São poucos os trabalhos que tratam problemas estruturados em DAG, utilizando a abordagem de classificação global. Além disso, existe o problema da classificação multirrótulo, na qual uma instância pode ser predita como pertencente a mais de uma classe simultaneamente.

Dessa maneira, esta pesquisa busca desenvolver um classificador hierárquico multirrótulo para estrutura DAG que utiliza como técnica da aprendizagem uma rede neural artificial. Redes neurais já foram utilizadas para tratar de problemas hierárquicos estruturados em árvore, mas não em DAG.

Esse classificador utiliza a abordagem de classificação global capaz de avaliar a hierarquia em um único passo preditivo a qual poderá ocorrer em qualquer nó da hierarquia, seja em nós folha ou não.

Outra questão refere-se ao tipo de aprendizagem da rede neural que será utilizado. A aprendizagem será realizada usando uma rede neural competitiva baseada no algoritmo LVQ (*Learning Vector Quantization).*

Apesar de o classificador ter sido desenvolvido para estrutura DAG, nada impede que seja utilizado em dados estruturados na forma de árvore.

Além da utilização da técnica neural para o treinamento na rede também será utilizada a técnica de estratégia evolucionária.

1.4 CONTRIBUIÇÕES

Uma das importantes contribuições desta tese é o desenvolvimento de um classificador multirrótulo hierárquico baseado em uma rede neural competitiva, chamado de MHC-CNN (Multi-label Hierarchical Classification using a Competitive Neural Network).

Até onde se sabe, alguns classificadores usando redes neurais foram desenvolvidos para tratar de problemas hierárquicos, porém todos destinados a classificação estruturada em árvore. Sendo assim, o MHC-CNN pode ser considerado o primeiro classificador hierárquico que utiliza uma rede neural para tratar problemas do tipo DAG.

Além disso, também é o primeiro classificador hierárquico que utiliza como técnica de aprendizagem da rede neural a aprendizagem competitiva.

Outra grande contribuição é o fato de se utilizar a abordagem de classificação global no desenvolvimento do classificador hierárquico, o qual discrimina todas as classes do domínio em um único passo preditivo.

Um detalhe a ser observado e considerado é que o classificador MHC-CNN pode ser utilizado também em estruturas em árvore.

Uma segunda versão do classificador MHC-CNN, denominada de MHC-ES é desenvolvida a qual utiliza como técnica de aprendizagem da rede neural a estratégia evolucionária.

1.5 ORGANIZAÇÃO DO TRABALHO

Este documento está organizado em oito capítulos. O Capítulo 2 apresenta o embasamento teórico sobre a classificação de dados. O Capitulo 3 descreve a classificação hierárquica bem como a contextualização em que esse trabalho se insere. O Capítulo 4 apresenta os trabalhos correlatos referentes ao tema desta tese. O Capítulo 5 apresenta o algoritmo proposto, MHC-CNN, para tratar problemas de classificação hierárquica multirrótulo, bem com uma segunda versão do algoritmo MHC-CNN, denominada de MHC-ES, o qual utiliza uma técnica alternativa para o treinamento do classificador. O Capítulo 6 apresenta os experimentos e os resultados obtidos. E por fim, o Capítulo 7 relata as conclusões obtidas e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste Capítulo, são apresentadas informações sobre a tarefa de classificação. Na Seção 2.1 citam-se os principais conceitos referentes à classificação de dados. Na Seção 2.2 são descritas algumas técnicas utilizadas na classificação de dados, tais como redes neurais artificiais e estratégias evolucionárias, cujas técnicas são utilizadas neste trabalho. Por fim, a Seção 2.3 descreve as principais medidas utilizadas para a avaliação dos classificadores.

2.1 PRINCIPAIS CONCEITOS DE CLASSIFICAÇÃO

A classificação é um processo que consiste em associar uma determinada instância (exemplo) a uma ou mais classes, dentre um conjunto de classes previamente definidas. Essa associação de um exemplo a uma determinada classe ocorre conforme as características (atributos) de cada instância.

Em aprendizagem de máquina, a classificação faz parte de um tipo aprendizagem denominada de aprendizagem supervisionada (MITCHELL, 1997), a qual consiste em construir um algoritmo de indução capaz de encontrar um bom classificador a partir de um conjunto de exemplos rotulados. A Figura 2 ilustra esse processo de classificação. Primeiramente, os dados brutos são preparados em um conjunto de exemplos para que possam ser processados. Um conjunto de exemplos é composto por valores de atributos, que são características do exemplo, e pelo atributo classe. Nessa figura é mostrado o formato padrão de um conjunto de exemplos T com m exemplos e n atributos. A linha i refere-se ao i-ésimo exemplo onde $i = 1,2,...,n$ e a entrada x_{ij} refere-se ao valor do j-ésimo atributo X_j do exemplo i, onde $j = 1,2,...,m$ (REZENDE, 2005).

Após o processamento dos dados, esse conjunto de exemplos será submetido à entrada do algoritmo de indução para que seja feito o treinamento do classificador. O objetivo do treinamento é encontrar uma função que mapeie cada exemplo T_i com a sua classe y_i correspondente.

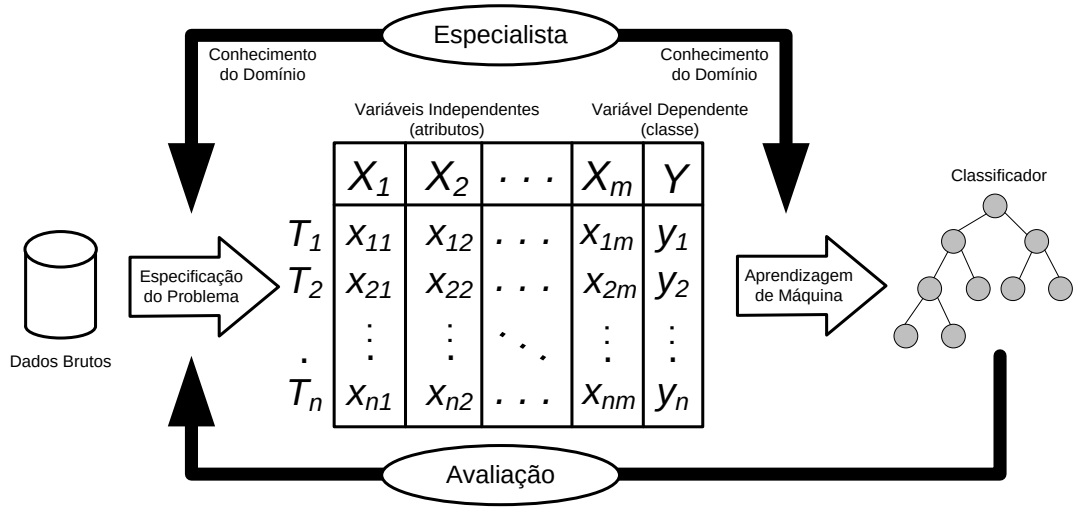


Figura 2 - Processo de Classificação (REZENDE, 2005).

Após a etapa de treinamento tem-se um classificador que deve ser capaz de predizer corretamente o rótulo de novos exemplos, que ainda não foram rotulados (REZENDE, 2005). Um exemplo dessa etapa é mostrado na Figura 3. Observa-se o conjunto de dados formado por 8 amostras as quais são submetidas a um algoritmo de aprendizagem baseado em árvore de decisão. Ao final desse treinamento um modelo que descreve o conjunto de classes é gerado pelo algoritmo. Esse modelo será utilizado na etapa de teste, onde amostras desconhecidas são submetidas ao modelo e esse deverá ser capaz de predizer as classes dessas amostras. Os dados usados para exemplificar a etapa de treinamento e de teste foram retirados da base de dados Iris, disponível com o software WEKA, e o algoritmo de classificação utilizado foi J48.

Em geral, a classificação pode ser dividida em dois tipos: a classificação plana (convencional) e a classificação hierárquica.

2.2 TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA A CLASSIFICAÇÃO DE DADOS

Diversos são os paradigmas de aprendizado que determinam a abordagem a ser utilizada pela técnica de AM para a realização do processo de indução. Dentre elas tem-se o paradigma Conexionista, cuja principal técnica são as Redes Neurais Artificiais e o paradigma Evolutivo, tendo como uma das técnicas as Estratégias Evolucionárias.

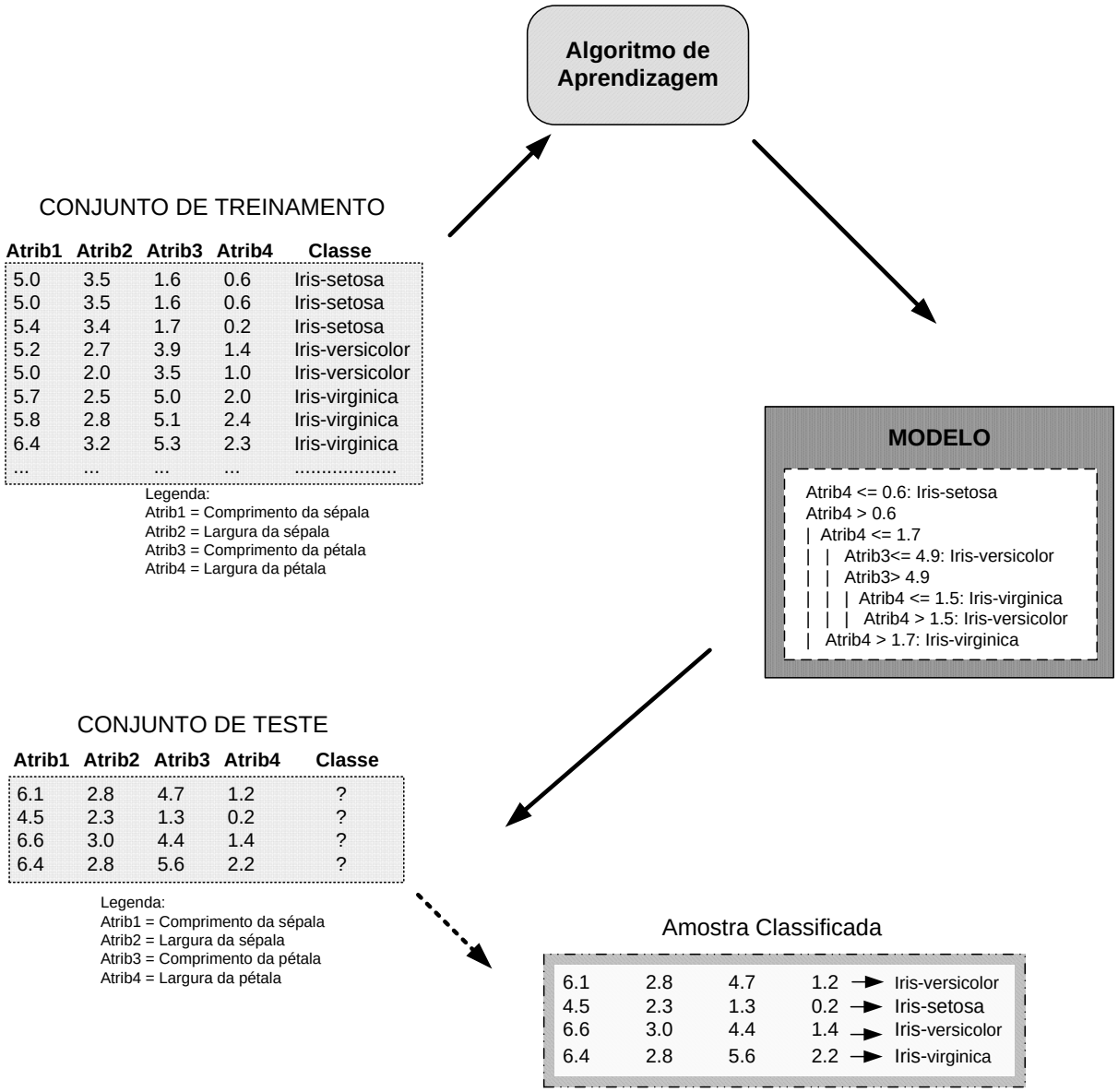


Figura 3 – Etapa de Treinamento e Teste de um Classificador.

2.2.1 Redes Neurais Artificiais

Redes Neurais Artificiais (RNA) são modelos matemáticos inspirados nas estruturas biológicas que compõe o cérebro humano e que possuem a capacidade computacional adquirida por meio de aprendizado e generalização (HAYKIN, 2001).

Uma das definições mais clássicas para RNA foi proposta por Haykin (2001), conceituando-as da seguinte forma:

> "Uma Rede Neural é um processador maciçamente paralelamente distribuído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso".

Também segundo Haykin (2001), uma RNA se assemelha ao cérebro em dois aspectos:

> "1- O conhecimento adquirido pela rede, a partir de seu ambiente, através de um processo de aprendizagem. 2- Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizados para armazenar o conhecimento adquirido".

Para que ocorra o aprendizado em uma rede neural faz-se necessário o uso de um algoritmo de aprendizagem, o qual terá a função de adaptar os pesos sinápticos da rede de uma forma ordenada.

Uma RNA é formada por diversos neurônios que interagem entre si através de unidades estruturais e funcionais denominadas de sinapses.

Um neurônio é uma unidade de processamento de informação que é fundamental para a operação de uma RNA. A Figura 4 ilustra o modelo de um neurônio.

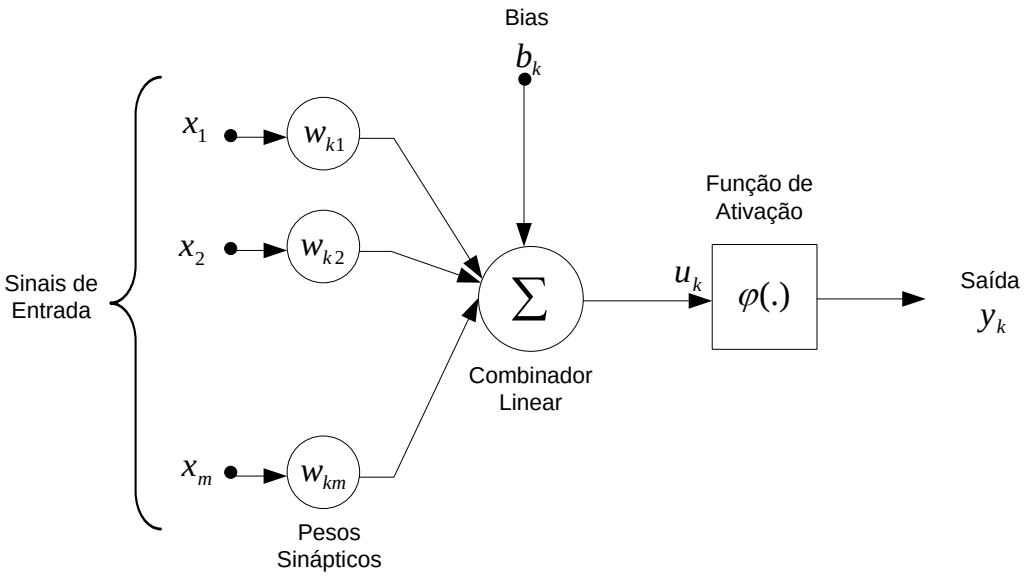


Figura 4 – Modelo Não-Linear de um Neurônio (HAYKIN, 2001).

Esse modelo consiste de:

1º - Um conjunto de sinapses, cada uma caracterizada por um peso próprio. Especificamente, um sinal de entrada x_i na entrada da sinapse j conectada ao neurônio k é multiplicado pelo peso sináptico w_{kj} .

2º - Um combinador linear para somar os sinais de entrada, ponderados pela respectiva sinapse do neurônio.

3º - Uma função de ativação para restringir a amplitude da saída de um neurônio. Essa função limita a faixa de amplitude permitida, a qual, normalmente, é limitada ao intervalo fechado de [0,1] ou, alternativamente de [-1,1].

Além desses três itens, um modelo neuronal inclui também um bias aplicado externamente por b_k. Esse bias tem como objetivo aumentar (se for positivo) ou diminuir (se for negativo) a entrada total da função de ativação.

Matematicamente, pode-se definir um neurônio k através da Equação 1 e Equação 2.

$$u_k = \sum_{j=0}^{m} w_{kj} x_j \qquad \textbf{Equação 1}$$

$$y_k = \varphi(u_k + b_k) \qquad \textbf{Equação 2}$$

em que $x_1, x_2, ..., x_m$ são os atributos de entrada, $w_{k1}, w_{w2}, ..., w_{km}$ são os pesos sinápticos do neurônio k, u_k é a saída do combinador linear, b_k é o peso ligado ao bias, $\varphi(.)$ é a função de ativação e y_k é o sinal de saída do neurônio.

O uso do bias tem o efeito de aplicar uma transformação à saída u_k do combinador linear, conforma mostra a Equação 3.

$$v_k = u_k + b_k \qquad \textbf{Equação 3}$$

Arquitetura das RNA

A maneira como os neurônios estão estruturados está intimamente ligada com o algoritmo de aprendizagem usado para treinar a rede. Conforme Haykin (2001) existem três classes funcionais de arquitetura de rede neural, que são: Rede Neural de Uma Camada (*Single-Layer Feedforward*), Rede Neural Multicamadas (*Multi-Layer Feedforward*) e Redes Recorrentes (*Recurrent Network*). Como o foco desse trabalho é a utilização de uma Rede Neural de uma Camada na sequência são apresentados maiores detalhes desta arquitetura.

A arquitetura de uma rede neural de uma camada é a forma mais simples de camadas de redes em que os neurônios estão organizados em camadas (HAYKIN, 2001). Nessa arquitetura tem-se a camada de entrada, constituída de atributos de entrada, a qual nem sempre é contada como uma camada, pois os neurônios não executam nenhuma função,

porém ela está conectada com a camada de saída ou de processamento, formada por neurônios.

A Figura 5 ilustra um exemplo dessa arquitetura de rede. Observa-se nesta figura que há cinco atributos de entrada representados por $x_1, x_2, ..., x_5$ na camada de entrada, e três neurônios na saída, representados por n_1, n_2, n_3. Cada atributo (neurônio) de entrada está conectado com cada neurônio da camada de saída. Para cada ligação existe um peso w_{kj} associado.

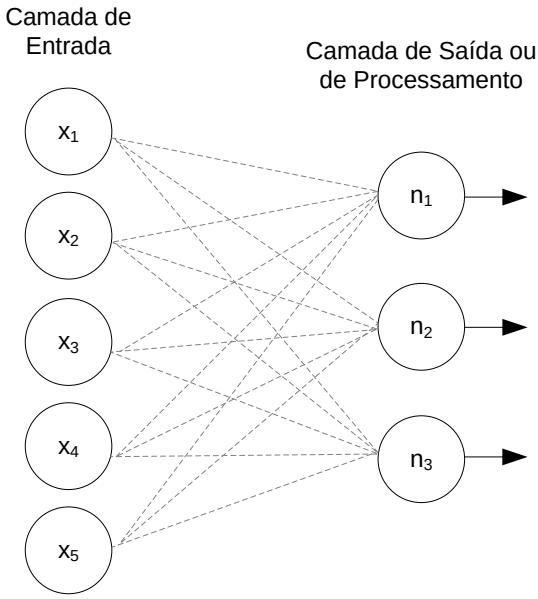


Figura 5 – Exemplo de uma RNA de Uma Camada (HAYKIN, 2001).

Nota-se ainda nessa arquitetura que não há conexões no sentido da camada de saída para a camada de entrada.

Métodos de Aprendizagem das RNA

O aprendizado de uma RNA ocorre através de um processo iterativo de ajustes de pesos sinápticos através de alguma função pré-estabelecida.

De acordo com Haykin (2001), a aprendizagem de uma RNA pode ser definida como:

> "Aprendizagem é um processo pelo qual os parâmetros livres de uma rede neural são adaptados através de um processo de estimulação pelo ambiente no qual a rede está inserida. O tipo de aprendizagem é determinado pela maneira pela qual as modificações dos parâmetros ocorrem".

Sendo assim, o processo de aprendizagem implica em uma sequência de eventos, no qual a RNA precisa ser estimulada por um ambiente, sofrer modificações nos seus parâmetros, e responder de uma maneira nova ao ambiente, devido às modificações ocorrida na sua estrutura interna (HAYKIN, 2001). Esses eventos são implementados em um algoritmo de aprendizagem, o qual deve seguir algumas regras de aprendizagem.

Existem diversos métodos desenvolvidos para o treinamento das redes, podendo ser agrupados em dois tipos: aprendizado supervisionado e aprendizado não supervisionado.

No aprendizado supervisionado há a presença de um supervisor que realiza o papel de monitorar a resposta da rede. Nesse tipo de aprendizagem, o conjunto de treinamento é formado por pares de entrada e saída (x_i, y_i^d), em que x_i representada um padrão de entrada e y_i^d representa a saída desejada para o padrão fornecido. O ajuste dos pesos w_{kj} é feito de maneira que a resposta y_i da rede para o padrão x_i se aproxime da saída y_i^d desejada.

No aprendizado não supervisionado não existe a presença do supervisor. O treinamento é formado pelos padrões de entrada x_i e o ajuste dos pesos w_{kj} é obtido através de valores do padrão de entrada.

Haykin (2001) apresenta cinco regras básicas de aprendizagem de RNA: Aprendizagem por Correção de Erro, Aprendizagem Baseada em Memória, Aprendizagem Hebbiana, Aprendizagem Competitiva e Aprendizagem Boltzmann. O classificador desenvolvido utiliza como técnica para o treinamento da rede neural a aprendizagem competitiva, por isso, será detalhado o funcionamento deste tipo de aprendizagem.

2.2.2 Aprendizagem Competitiva

A aprendizagem competitiva, como o próprio nome sugere, consiste em uma disputa (competição) entre os neurônios da camada de saída, da qual apenas um neurônio sairá como vencedor.

A rede básica do aprendizado competitivo tem uma camada de neurônios de saída, também chamada de camada competitiva ou de processamento, onde cada neurônio dessa camada está totalmente conectado aos nós de entrada. Cada neurônio da camada de entrada está conectado com todos os neurônios na camada competitiva. A saída com maior ativação inicial terá maior chance de vencer a disputa com as outras saídas e estas perderão o poder de inibição ao longo do tempo sobre a saída de maior ativação, fazendo com que fiquem

completamente inativas, exceto a vencedora. Este método é conhecido como Winner-Takes-All.

O objetivo da aprendizagem competitiva é fazer com que neurônios se especializem em estímulos apresentados de forma não supervisionada, isto é, nenhuma informação sobre a classe do estímulo apresentado é usada no processo de ajuste dos pesos sinápticos. Todos os neurônios recebem o mesmo conjunto de entradas e competem, através de uma dinâmica que usa conexões laterais, com todos os outros neurônios. Estas conexões laterais podem ser positivas (no caso da auto-realimentação) ou inibitórias (negativas) (HAYKIN, 2001).

Na camada competitiva, os neurônios competem pela oportunidade de responder aos padrões de entrada. O neurônio vencedor representa a categoria de classificação para o padrão de entrada selecionado.

Entre as redes que utilizam a aprendizagem competitiva no processo de aprendizagem podem-se citar as rede auto-organizáveis e as redes com quantização do vetor de aprendizagem (LVQ – Learning Vector Quantization). Os modelos de redes auto-organizáveis conhecidos que utilizam o aprendizado competitivo são os modelos da família Adaptive Resonance Theory (ART) e o modelo Self-organizing Map (SOM) que são utilizados para o agrupamento de dados. Já a LVQ é utilizada para a classificação de dados. Na sequência será explicado o funcionamento das redes SOM e LVQ.

2.2.2.1 Rede Self-Organizing Map

A rede Self-Organizing Map (SOM), também chamada de rede de Kohonen, foi desenvolvida por Teuvo Kohonen em 1982 (KOHONEN, 1990) e o objetivo principal é transformar um padrão de sinal incidente de dimensão arbitrária em um mapa discreto uni- ou bidimensional e realizar esta transformação adaptativamente de uma maneira topologicamente ordenada (HAYKIN, 2001).

O algoritmo de aprendizagem tem natureza local e as modificações dos pesos sinápticos são confinadas à vizinhança do neurônio ativado. A ordem global, ou seja, o equilíbrio da rede surge de interações locais. Os neurônios de saída competem entre si para serem ativados, de forma que apenas um neurônio de saída seja considerado “vencedor”.

Os neurônios em uma rede de Kohonen são colocados nos nós de uma grade (que apresenta uma topologia particular, que pode ser retangular, hexagonal etc.) que é usualmente de uma ou duas dimensões. Mapas de dimensões maiores são também possíveis, porém de

mais difícil aplicação e compreensão. Cada neurônio na grade é completamente conectado a todos os neurônios da camada de entrada (HAYKIN, 2001). São apresentados os padrões de entrada à rede, e a cada padrão apresentado tem-se uma região de atividade na grade. A localização e natureza de uma determinada região variam de um padrão de entrada para outro. Assim sendo, todos os neurônios da rede devem ser expostos a um número suficiente de diferentes padrões de entrada, garantindo assim que o processo de auto-organização ocorra de forma apropriada.

Serão considerados “vencedores” os neurônios que mais se assemelharem ao padrão de entrada, sendo que para esta comparação são utilizadas medidas de distâncias (normalmente utiliza-se a distância euclidiana).

Primeiramente, deve-se determinar a taxa de aprendizagem e o tamanho do raio topológico que decrescem conforme o treinamento é executado, bem como a topologia da camada de saída, ou seja, o critério a ser utilizado para determinar quais neurônios pertencem a vizinhança do neurônio vencedor (por exemplo: hexagonal, retangular etc.).

O algoritmo responsável pela formação do mapa de Kohonen primeiramente inicializa os pesos sinápticos da rede. Este procedimento pode ser feito atribuindo pequenos valores aleatórios, de igual grandeza aos padrões de entrada; fazendo desta forma, nenhuma organização prévia é imposta ao mapa de características (HAYKIN, 2001).

Após as inicializações, o algoritmo, em sua fase de treinamento, passa por três etapas básicas: competição, cooperação e adaptação sináptica.

Na primeira fase, competição, para cada padrão de entrada, os neurônios do mapa calculam seus respectivos valores de uma função discriminante. Esta função fornece a base para a competição entre os neurônios. O neurônio com o maior valor da função discriminante é declarado vencedor da competição.

Na fase seguinte, cooperação, o neurônio vencedor determina a localização espacial de uma vizinhança topológica de neurônios excitados que cooperarão entre si.

A última fase, adaptação sináptica, os neurônios excitados aumentam seus valores individuais da função discriminante em relação ao padrão de entrada através de ajustes adequados aplicados a seus pesos sinápticos. Os ajustes feitos são tais que a resposta do neurônio vencedor à aplicação subsequente de um padrão de entrada similar é melhorada. A Tabela 1 apresenta a descrição do algoritmo SOM.

Tabela 1 – Passos do Algoritmo SOM (HAYKIN, 2001).

Entrada de Dados

- Conjunto de exemplos de entrada, em que cada exemplo é dado por $x_i = [a_1, a_2, ..., a_m]$

Passo 1: Inicialização

- Taxa de Aprendizagem α
- Definir a topologia de rede t
- Raio Topológico σ
- Número de Épocas e
- Criar a Rede Neural RN de tamanho $l \, x \, c$ onde l representa a quantidade de linhas e c a quantidade de colunas do mapa. A RN é constituída por neurônios w_{ij}, onde $w_{ij} = [b_1, b_2, ..., b_k]$. Cada elemento de w_{ij} normalmente é gerado com valores aleatórios entre 0 e 1 e $k = m$.

Passo 2: Critério de Parada

- Comparar o número máximo de épocas e/ou a alteração dos pesos da rede neural.

Passo 3: Treinamento

- Para cada exemplo de entrada x_i
- Fase Competitiva
  - Calcular as distâncias para cada exemplo de entrada com cada neurônio da rede neural. Um exemplo de distância que pode ser usada é a distância Euclidiana $d_{k=1}^{j} = \sum_{i=1}^{m} \sqrt{(x_i - w_{ij})^2}$
  - Definir o neurônio que apresentar a menor distância como vencedor: $n(v) = \min(d_{k=1}^{j})$
- Fase Cooperativa
  - Localizar os vizinhos do neurônio vencedor $n(v)$ conforme t e σ. Se o neurônio analisado fizer parte da vizinhança $z(l) = 1$, caso contrário $z(l) = 0$ ($z(l)$ guardará os vizinhos que deverão ser atualizados).
- Fase Adaptativa
  - Atualizar a rede neural conforme: $w_{ij}(t+1) = w_{ij}(t) + a * (x_i - w_{ij}(t)) * z(l)$.
- Retornar ao Passo 2 e se esse continuar sendo válido repetir o Passo 3 e o Passo 4.

Passo 4: Atualização

- Atualizar a Taxa de Aprendizagem α usando uma função linear, exponencial ou geométrica em função do número de épocas e.
- Atualizar o Raio Topológico (definido como uma função monotonicamente decrescente em função das épocas).

Saída de Dados

- Conjunto de Pesos Considerado Adequado.

2.2.2.2 Rede Learning Vector Quantization

A quantização vetorial por aprendizagem (LVQ) é uma técnica de aprendizagem supervisionada que usa a informação das classes para mover ligeiramente os vetores de

Voronoi (também chamados de vizinhos mais próximos baseada na métrica euclidiana). No caso da rede neural serão os vetores de pesos sinápticos dos neurônios, a fim de melhorar a qualidade das regiões de decisão do classificador.

O algoritmo SOM fornece um método aproximado para calcular os vetores de Voronoi, de uma maneira não-supervisionada, com a aproximação sendo especificada pelos vetores de pesos sinápticos dos neurônios no mapa de características (HAYKIN, 2001). Dessa forma, o cálculo do mapa de característica pode ser visto como uma de duas etapas (ver Figura 6) para resolver de forma adaptativa o problema de classificação de padrões. A segunda etapa é feita pela quantização vetorial por aprendizagem, a qual fornece um mecanismo para o ajuste fino do mapa de características.

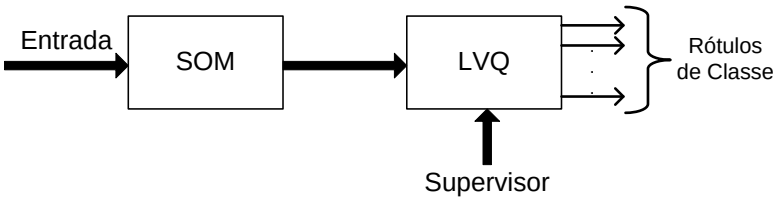


Figura 6 – Esquema de classificação adaptativa de padrões (HAYKIN, 2001).

Suponha que $\{w_{ij}\}_{j=1}^{t}$ representa o conjunto de vetores de Voronoi e que $\{x_i\}_{i=1}^{N}$ representa o conjunto de vetores de entrada. O algoritmo LVQ funciona da seguinte maneira:

1º - Um vetor de entrada x_i é selecionado aleatoriamente. Se os rótulos de classe do vetor de entrada x_i e de um vetor de Voronoi w_{ij} concordarem, ou seja, as classes forem iguais, o vetor de Voronoi w_{ij} é movido em direção ao vetor de entrada x_i (Figura 7a), caso contrário, se eles discordarem, ou seja, as classes forem diferentes, o vetor de Voronoi w_{ij} é afastado do vetor de entrada x_i (Figura 7b).

2º - Através da comparação feita no item anterior será possível atualizar os vetores de Voronoi, ou seja, os pesos sinápticos do neurônio vencedor.

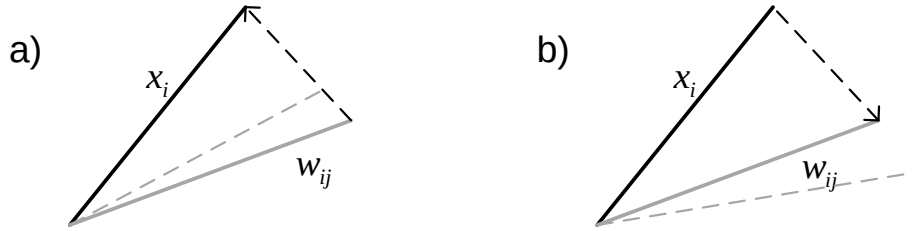


Figura 7 - Atualização dos vetores de Voronoi nas redes LVQ (HAYKIN, 2001).

3º - É necessário que a constante de aprendizagem $a(t)$ decresça monotonicamente com o número de iterações t.

Kohonen apresentou três versões diferentes do algoritmo LVQ, que chamou de LVQ1, LVQ2 e LVQ3.

Na versão LVQ1 procura-se encontrar um conjunto de pesos que minimize o erro de classificação para o classificador k-Nearest Neighbor (k-NN) com k=1. A Equação 4 mostra o cálculo para o ajuste dos pesos desta versão.

$$w_{ij}(t+1) = \begin{cases} w_{ij}(t) + a(t) * (x_i(t) - w_{ij}(t)), \textit{ se a classe for correta} \\ w_{ij}(t) - a(t) * (x_i(t) - w_{ij}(t)), \textit{ caso contrário} \end{cases} \quad \textbf{Equação 4}$$

em que $a(t)$ é a taxa de aprendizagem.

Já a versão LVQ2 a decisão de classificação é parecida com a versão anterior, porém baseia-se na ideia de deslocamento dos vetores utilizando conceitos Bayesianos. Nesta versão consideram-se dois vetores w_{i1j} e w_{i2j} (neurônios) que são os vizinhos mais próximos do exemplo x_i, e que serão atualizados simultaneamente. Um dos vetores deve pertencer a classe correta e o outro a uma classe incorreta. Além disso, x_i deve cair em uma zona de valores denominada de janela λ que é definida em torno dos dois vetores. Então x_i é definido a cair na janela se $\min\left(\frac{d_{i1}}{d_{i2}}, \frac{d_{i2}}{d_{i1}}\right) \geq \frac{1-\lambda}{1+\lambda}$, em que d_{i1} e d_{i2} representam as distâncias do exemplo x_i com os vetores w_{i1j} e w_{i2j}. Assim, as atualizações dos pesos dos neurônios na iteração t são mostradas na Equação 5 (KOHONEN, 1995).

$$\begin{aligned} w_{i1j}(t+1) &= w_{i1j}(t) - a(t) * (x_i(t) - w_{i1j}(t)), \textit{ se a classe for incorreta} \\ w_{i2j}(t+1) &= w_{i2j}(t) + a(t) * (x_i(t) - w_{i2j}(t)), \textit{ se a classe for correta} \end{aligned} \quad \textbf{Equação 5}$$

Na versão LVQ3 foi inserida uma correção na versão LVQ2, correção essa que assegura que w_{ij} continue se aproximando das distribuições de classes (KOHONEN, 1995). Assim, além das verificações feitas na versão LVQ2 e as atualizações apresentadas através da Equação 5 faz-se outro teste. Enquanto x_i não estiver dentro da janela λ, a atualização dos pesos dos neurônios se dará pela fórmula mostrada na Equação 6.

$$w_{ikj}(t+1) = w_{ikj}(t) - \varphi a(t) * (x_i(t) - w_{ikj}(t)), \textit{ se } \varphi \in \{i, j\} \textit{ e todas as classes forem corretas} \quad \textbf{Equação 6}$$

Os valores recomendados de λ devem estar compreendidos entre 0.2 e 0.3, e de φ deve situar-se entre 0.1 e 0.5 (KOHONEN, 1995).

2.2.3 Estratégias Evolucionárias

As Estratégias Evolucionárias (EEs) são uma das técnicas da Computação Evolucionária que propõe um paradigma para a solução de problemas inspirado na seleção natural (DARWIN, 1859).

As EEs foram desenvolvidas por RECHENBERG (1965), SCHWEFEL (1975) na Universidade Técnica de Berlim por volta de 1964. É um algoritmo em que indivíduos (soluções potenciais) são codificados por um conjunto de variáveis de valores reais, o "genoma" (KUSIAK, 2000). As EEs foram inicialmente desenvolvidas com o propósito de otimização de parâmetros.

As EE utilizam mutações normalmente distribuídas para modificar valores reais. Os principais operadores são mutação e cruzamento. Já o operador de seleção utilizado é determinístico e o tamanho da população de pais geralmente difere do tamanho da população de filhos.

Há quatro tipos de estratégias evolutivas tais como: Dois-membros: (1 + 1)-EE, Multimembros: (μ + 1)-EE, Multimembros: (μ + λ)-EE e Multimembros: (μ , λ)-EE.

A primeira versão algoritmo de EE utilizava um esquema mutação e seleção conhecido como Dois-membros (*two-membered*) EE ou (1 + 1)-EE. Já população é formada por dois indivíduos, um pai e um descendente ou filho. O descendente é gerado por meio da mutação e o melhor dos dois se torna pai na próxima geração. O operador de mutação adiciona valores gerados conforme uma distribuição com média zero e desvio padrão definido para cada atributo do vetor pai.

A segunda versão do algoritmo denominada de (μ + 1)-EE, proposta por Rechenberg (1973), permite que mais de um pai participe da geração de um filho. Com a introdução de *m* pais, a reprodução sexuada torna-se possível via um operador de recombinação *r* que é aplicado aos dois vetores que compõem um indivíduo da população. Este operador é denominado de operador de recombinação discreto e é equivalente ao *crossover* uniforme em algoritmos genéticos (BÄCK et al., 2000). Dessa forma, todos os pais da população possuem a mesma probabilidade de reprodução. O operador de seleção remove o indivíduo menos apto (com menor *fitness*) dentre os pais e o filho gerado e os demais formam a população da próxima geração.

Uma terceira versão denominada de (μ + λ)-EE foi proposta inicialmente, onde λ pais produzem λ filhos e a população μ + λ é posteriormente reduzida para μ indivíduos. A seleção

opera no conjunto união de pais e filhos. Assim, os pais sobrevivem até que filhos com *fitness* superiores a eles sejam gerados.

Conforme Schwefeel (1975) esta característica pode levar a alguns problemas, tais como: em problemas com superfície de *fitness* dinâmicas (que variam ao longo do tempo) a estratégia pode ficar presa em um ótimo que não é mais um ótimo da superfície de *fitness* atual. Isso também pode ocorrer na presença de ruído.

Para tentar solucionar esse problema Schwefeel (1995) criou uma estratégia denominada de (μ , λ)-EE, onde somente os filhos sofrem seleção, restringindo o período de vida de cada indivíduo a uma geração.

Em resumo, as duas últimas estratégias, (μ + λ)-EE e (μ , λ)-EE, possuem a mesma estrutura. A principal diferença entre as diversas variantes das estratégias evolutivas está na forma de atualização do vetor de parâmetros σ.

Uma das características mais importantes das Estratégias Evolucionárias é a auto-adaptação de parâmetros da estratégia (BÄCK et al 2000). Esse termo refere-se a parâmetros que controlam o processo de busca evolucionário, tais como taxas de mutação, variâncias de mutação e probabilidades de recombinação.

No caso mais geral de uma estratégia evolutiva (μ , λ), um indivíduo $v = (x, \sigma, \theta)$ pode ser composto por três componentes: x é o vetor de atributos onde $x \in \Re^l$, σ é o vetor de desvios padrões das mutações onde $\sigma \in \Re^{l\sigma}$ e θ é o vetor dos ângulos de rotação onde $\theta \in (0,2]^{l\theta}$, l é a dimensão de x, $l\sigma \in \{1,...,l\}$, e $\theta l \in \{0,(2l - l\sigma)(l\sigma - 1)/2\}$ (BÄCK et al 2000).

O operador de mutação trabalha adicionando uma variável randômica X de dimensão n, com distribuição normal de acordo com cada um dos parâmetros de estratégia σ' e θ' do indivíduo (BÄCK et al 2000).

Dependendo do número de parâmetros da estratégia incorporados na representação de um indivíduo, os seguintes tipos de mutação podem ser citados:

a) Único σ' para todos os atributos v_i do indivíduo (BÄCK et al 2000) conforme ilustrada a Equação 7.

$\sigma' = \sigma \exp(\tau_0 N(0,1))$, sendo $\tau_0 \propto n^{-1/2}$ **Equação 7**

$v_i = v_i + \sigma' N(0,1)$

na qual n é o número de atributos e τ_0 é uma taxa de aprendizado configurável.

b) Cada atributo v_i do indivíduo tem seu σ_i (BÄCK et al 2000) conforme ilustrada a Equação 8 .

$\sigma' = \sigma \exp(\tau' N(0,1) + \tau N(0,1))$, sendo $\tau' \propto (2n)^{-1/2}$ e $\tau \propto (2n^{1/2})^{-1/2}$ **Equação 8**

$v_i = v_i + \sigma' N_i(0,1)$

em que n é o número de atributos e τ' e τ é são as taxas de aprendizagem.

c) Cada atributo v_i do indivíduo tem seu σ_i e ângulos de rotação θ_j as rotações de coordenada necessárias para transformar um vetor de mutação não correlacionado em um correlacionado. Conforme BÄCK et al (2000) a mutação é realizada conforme a ilustração da Equação 9.

$\sigma_i' = \sigma_i \exp(\tau' N(0,1) + \tau N_i(0,1))$, sendo $\tau' \propto (2n)^{-1/2}$ e $\tau \propto (2n^{1/2})^{-1/2}$ **Equação 9**

$\theta_j = \theta_j + \beta' N_j(0,1)$, sendo $\beta \approx 0.0873 \quad (5^o)$

$v_i' = v_i + N_i(0, C(\sigma', \theta'))$

com n sendo o número de atributos e τ' e τ as taxas de aprendizagem.

Em resumo, uma estratégia evolutiva pode ser implementada empregando-se a sequência de passos apresentada na Tabela 2.

Tabela 2 – Passos do Algoritmo de Estratégia Evolucionária (RECHENBERG, 1973).

Entrada de Dados
▪ $[P] = EE(\mu)$
▪ Quantidade de Gerações G
Passo 1: Inicialização
▪ μ indivíduos do tipo: $v = (x, \sigma, \theta)$
Passo 2: Critério de Parada
▪ Comparar o número máximo gerações.
Passo 3: Treinamento
▪ selecionar $p \geq 2$ pais aleatoriamente
▪ recombinar os indivíduos
▪ mutar os indivíduos
▪ avaliar os indivíduos gerados (filhos)
▪ repetir o Passo 3 λ vezes
▪ $G = G + 1$
▪ repetir o Passo 3
Saída de Dados
▪ μ melhores indivíduos de λ ou $\mu + \lambda$

2.3 AVALIAÇÃO DOS CLASSIFICADORES

Uma importante etapa no processo classificação é a avaliação dos algoritmos de classificação.

Para se obter a taxa de acerto e/ou a taxa de erro de um algoritmo é necessária medir a qualidade da classificação. Essas medidas são calculadas a partir dos exemplos que foram classificados corretamente e incorretamente, os quais são armazenados em uma matriz de confusão. Em um problema de classificação convencional binária quatro situações podem ocorrer. São:

1º – Verdadeiro Positivo (VP): O exemplo é predito corretamente como pertencendo a classe positiva.

2º – Falso Positivo (FP): O exemplo é predito como pertencendo à classe positiva, mas pertence à classe negativa.

3º – Verdadeiro Negativo (VN): O exemplo é predito corretamente como pertencente à classe negativa.

4º – Falso Negativo (FN): O exemplo é predito como pertencente à classe negativa, mas pertence à classe positiva.

A Tabela 3 ilustra a distribuição dessas quatro situações em uma matriz de confusão.

Tabela 3 – Matriz de Confusão para Classificação Convencional Binária

	Classe Predita	
Classe Verdadeira	**Positiva**	**Negativa**
Positiva	VP	FN
Negativa	FP	VN

Assim, a taxa de acerto (TA) e a taxa de erro (TE) de um classificador são obtidas conforme pode ser visualizada na Equação 10 e Equação 11, respectivamente (SUN & LIM, 2001).

$$TA = \frac{|VN| + |VP|}{|VN| + |VP| + |FP| + |FN|}$$ **Equação 10**

$$TE = \frac{|FN| + |FP|}{|VN| + |VP| + |FP| + |FN|} \text{ ou } TE = 1 - TA$$ **Equação 11**

Para avaliar a efetividade de um classificador para cada classe de um problema de classificação binária foram criadas as medidas de Sensibilidade (S), Especificidade (E) e Precisão (P).

A Sensibilidade (Equação 12) mede a capacidade de se predizer uma classe positiva cuja predição está correta, ou seja, ela indica quantos exemplos positivos foram previstos do total de exemplos. Essa medida também é conhecida como Revocação (*recall*) ou taxa de VP.

Equação 12

$$S = \frac{|VP|}{|VP| + |FN|}$$

A medida de Especificidade (Equação 13) mede a capacidade de se predizer uma classe negativa cuja predição está correta, ou seja, quantos exemplos negativos foram preditos do total de exemplos.

Equação 13

$$E = \frac{|VN|}{|VN| + |FP|}$$

A medida de Precisão (Equação 14) calcula a probabilidade da predição positiva estar correta em relação a todas as amostras.

Equação 14

$$P = \frac{|VP|}{|VP| + |FP|}$$

A medida de Precisão combinada com a medida de Sensibilidade origina a Medida-F (Equação 15). Essa realiza uma combinação balanceada das medidas de Precisão e Sensibilidade. Para esse balanceamento utiliza-se uma constante β que geralmente recebe valor 1 (SUN & LIM, 2001).

Equação 15

$$Medida - F = \frac{(\beta^2 + 1) * P * S}{\beta^2 * P + S}$$

A relação entre as medidas de Sensibilidade e Especificidade pode ser representada através da curva ROC (*Receiver Operating Characteristics*). A análise ROC teve a sua origem na teoria de detecção de sinais, para avaliar a qualidade de transmissão de sinal em um canal com ruído. Hoje é muito utilizada em técnicas de aprendizagem de máquina e mineração de dados como uma ferramenta para avaliação de modelos (FAWCETT, 2006).

Geometricamente, a curva é representada em um gráfico de pares de valores correspondentes a taxa de FP (1-E) e a taxa de VP (Sensibilidade), retratando a relação entre custo e beneficio, respectivamente, da predição de um exemplo classificado como positivo (FAWCETT, 2006).

Algoritmos que fornecem um valor contínuo, entre 0 e 1, para as suas predições, tal qual como as RNA, podem ser estabelecido um limiar para decidir se o exemplo pertence ou não a uma determinada classes. Assim, predições com valores acima do limiar são atribuídas a uma classe, caso contrário à outra. Cada limiar produzirá um conjunto distinto de pontos no espaço ROC. A Figura 8 mostra exemplos de curva ROC.

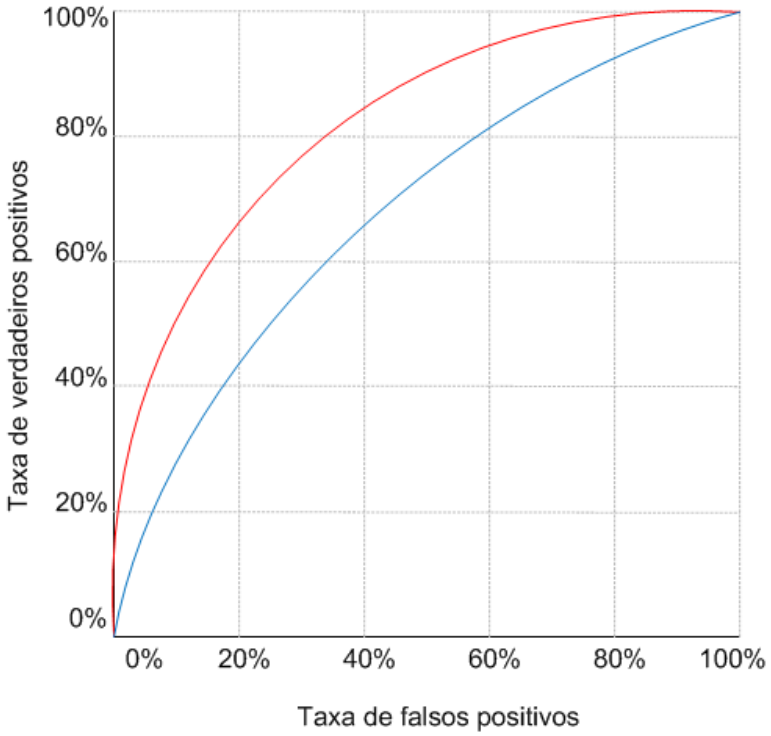


Figura 8 – Exemplos de Curva ROC (PRATI et al. 2008)

Nesse gráfico, a área abaixo da curva ROC é chamada de AUC (*Area Under the ROC Curve*). Essa área corresponde a uma medida quantitativa usada na comparação entre dois classificadores. Quanto mais próximo de 1 o valor da AUC, melhor é o desempenho do classificador.

3 CLASSIFICAÇÃO HIERÁRQUICA

Este Capítulo apresenta informações sobre a classificação hierárquica de dados. Na Seção 3.1 são apresentadas algumas definições sobre hierarquia. Na Seção 3.2 os conceitos fundamentais sobre classificação hierárquica são introduzidos, enquanto na Seção 3.3 são apresentadas as abordagens utilizadas para tratar problemas de classificação hierárquica. Já na Seção 3.4 são mostradas as medidas de avaliação utilizadas para avaliar modelos hierárquicos e por fim, na Seção 3.5 há uma breve descrição sobre as medidas estatísticas usadas na avaliação dos resultados.

3.1 DEFINIÇÕES

Uma hierarquia pode ser definida como ordenação de elementos seguindo a sua função ou importância. Geralmente uma forma de representar a hierarquia é através de um Grafo Acíclico Dirigido (Directed Acyclic Graph - DAG) conforme ilustra a Figura 9.

Definição de Grafo: Um grafo é uma estrutura $G(V, A)$ onde V é um conjunto não vazio de objetos denominados vértices ou nós e A é um conjunto de pares V não ordenados chamado arestas.

Definição de Grafo Acíclico Dirigido: Um grafo acíclico dirigido é um grafo dirigido, ou seja, cada aresta tem um sentido, sem ciclos.

Uma classe em uma hierarquia é representada como um vértice no grafo acíclico dirigido. As arestas do grafo mostram o relacionamento entre pai-filho: se p é pai de f, então existe uma aresta $(p, f) \in A$. Em geral, costuma-se omitir a direção das arestas em um grafo assumindo que ele é *top-down*.

Caminho em Grafo Acíclico Dirigido: Um caminho de um vértice v a um vértice y em um grafo $G(V, A)$ é uma sequência de vértices $v_0, v_1, v_2, ..., v_n$ tal que $v = v_0$ e $y = v_n$. O comprimento de um caminho é o número de arestas percorrido no caminho.

Nível de um Vértice em Grafo Acíclico Dirigido: O nível de um vértice em um grafo acíclico dirigido é o comprimento do menor caminho do vértice v até a raiz. Por exemplo, o nível do vértice I da Figura 9 é igual a 3.

Profundidade de Grafo Acíclico Dirigido: A profundidade de um grafo acíclico dirigido é a profundidade máxima de todos os vértices do grafo. No grafo da Figura 9, pode-se observar que a sua profundidade é igual a 5.

Relacionamento Hierárquico: Há vários tipos de relações entre os vértices da hierárquica, entre os mais conhecidos e utilizados pode-se citar: é-um, parte-de.

Ancestrais e Descendentes de um Vértice: O ancestral de um vertice v_n são todos os vértices pelos quais se passou quando percorrido um caminho, onde esse caminho pode ser representado por $v_0, v_1, v_2, ..., v_n$. Como exemplo, os nós ancestrais do nó J, na Figura 9, são os nós F, B e C. Já os descendentes de um vértice v_i são todos os nós, a partir dele, que estão conectados, até o seu nó folha. Como descendente do vértice J têm-se os nós O e N.

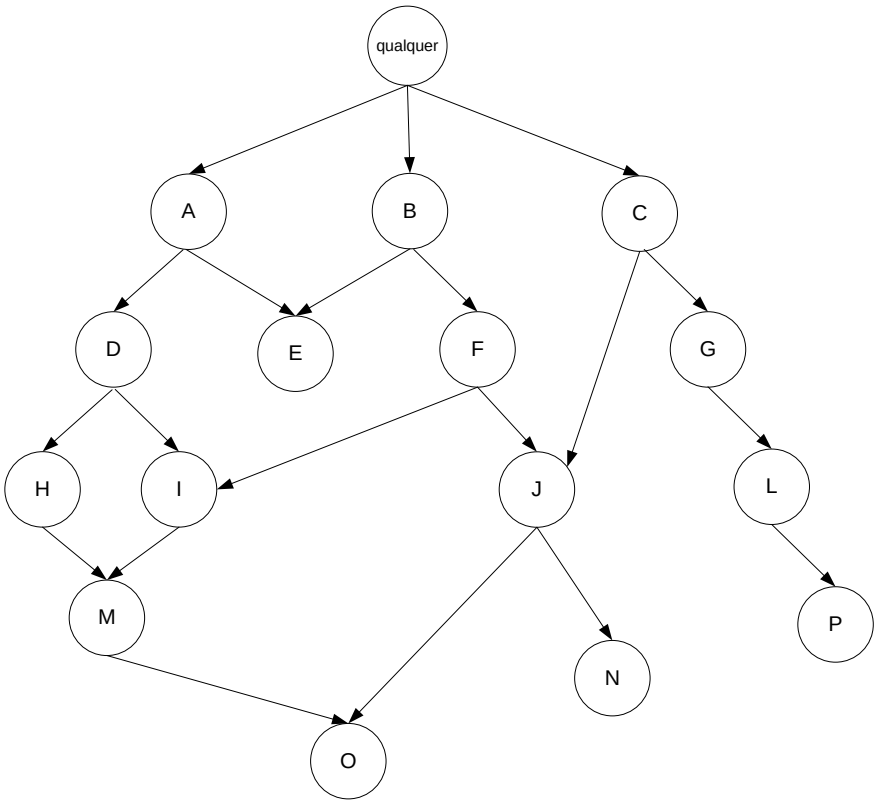


Figura 9 - Exemplo de um Grafo Acíclico Dirigido.

Um caso particular na teoria dos grafos são as árvores. Uma árvore $T(V, A)$ é um grafo simples conexo com $n-1$ arestas, em que n é o número de vértices. Em uma árvore um vértice está conectado com apenas um nó ancestral, conforme pode ser observado na Figura 10.

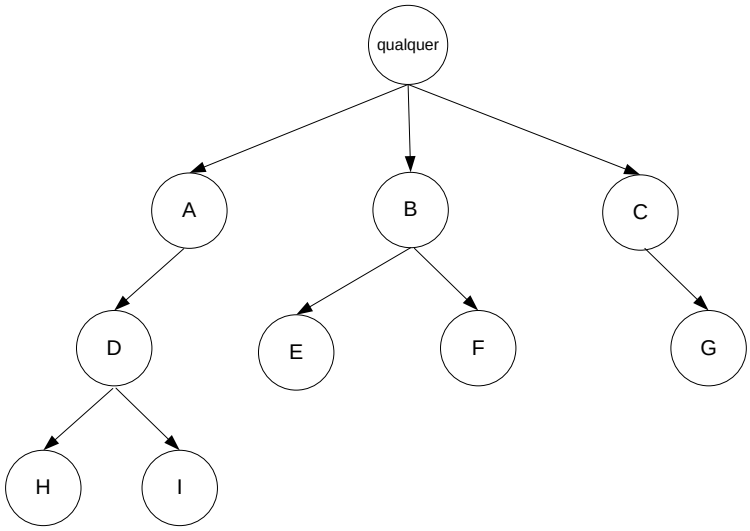


Figura 10 – Exemplo de uma árvore.

3.2 CONCEITOS FUNDAMENTAIS DE CLASSIFICAÇÃO HIERÁRQUICA

A classificação hierárquica, que é o tipo de classificação usada neste trabalho, difere da classificação convencional pelo fato das classes estarem dispostas em uma estrutura de hierarquia tal como um DAG ou uma árvore.

No processo de classificação hierárquica, como mostrado na Figura 11, deve-se ter o conjunto de dados, que será utilizado para treinamento do classificador e a hierarquia de classes que conterá as informações de ancestralidade e descendência das classes que serão preditas.

Neste tipo de classificação uma classe é um vértice dessa hierarquia. Assim, quando um exemplo de entrada prediz como sendo de uma determinada classe, automaticamente esse exemplo também será classificado como pertencentes a todas as suas classes ancestrais.

A classificação hierárquica difere da classificação convencional pelo fato das classes estarem dispostas em uma estrutura hierárquica. Em geral, as classes dessa estrutura podem estar dispostas na forma de uma árvore ou de um DAG. Essas estruturas são ilustradas nas Figura 12a e Figura 12b respectivamente. Nestas figuras, cada nó representa uma classe identificada por um número dentro do nó e as arestas (ligações entre os nós) representam as relações entre a classe ancestral e a classe descendente. Em ambas as figuras, o nó raiz corresponde a “qualquer classe”, denotando uma ausência total de conhecimento sobre a classe de um objeto.

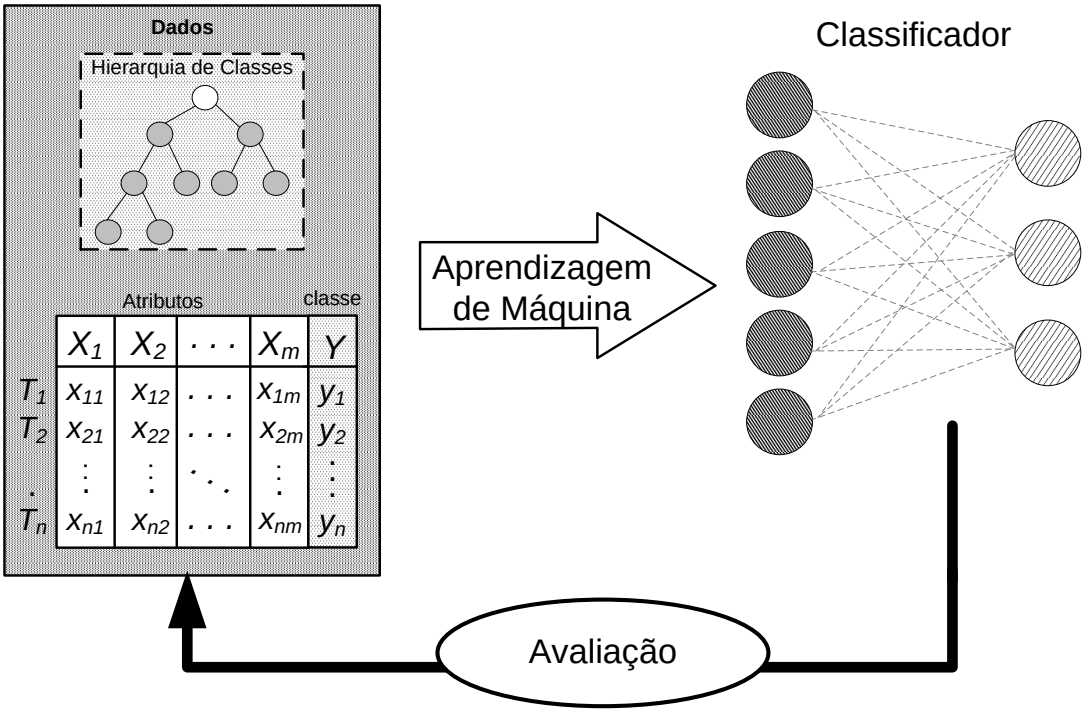


Figura 11 - Exemplo de do Processo de Classificação Hierárquica.

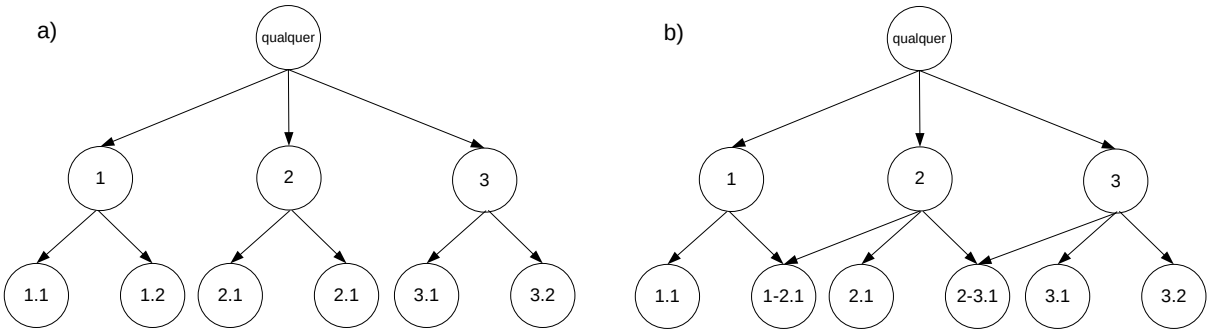


Figura 12 - Exemplo de hierarquia de classe estruturada em árvore e em DAG.

A principal diferença entre a estrutura de árvore e a estrutura de DAG é que na estrutura de árvore cada nó de classe, exceto o nó raiz, tem apenas um pai, enquanto que na estrutura de DAG cada nó de classe pode ter um ou mais nós pai.

Na estrutura de árvore, Figura 12a, cada nó é rotulado com o número da sua classe correspondente. Considera-se que o nó raiz está no nível 0 e o nível de qualquer outro nó é determinado pelo número de arestas que unem aquele nó ao nó raiz. Nota-se que os nós do primeiro nível têm um dígito, enquanto que nós do segundo nível têm dois dígitos: o primeiro

dígito identifica a classe pai (ao primeiro nível) e o segundo dígito identifica a subdivisão de classe ao segundo nível (FREITAS & CARVALHO, 2007).

A anotação para os rótulos dos nós de classe na Figura 12b é semelhante à anotação da Figura 12a. A diferença principal acontece para nós com múltiplos pais, isto é os nós, 1-2.1 e 2-3.1 no segundo nível da hierarquia de classe. Nesta notação, as classes ancestrais (pais) não são especificadas por um único dígito, mas por dois dígitos antes de delimitar o nível da classe com ".", por exemplo, um dígito para cada uma das classes pai. Consequentemente, na classe número 1-2.1, a anotação "1-2" indica que este nodo tem como pai as classes 1 e 2, e o "1" depois do "." indica que esta é a primeira classe descendente (filha) dessas duas classes ancestrais considerando como um todo (em lugar de cada classe ancestral individualmente) (FREITAS & CARVALHO, 2007).

Outra característica que difere a classificação hierárquica da classificação convencional refere-se ao tipo de predição das classes na hierarquia, as quais podem ser distinguidas em duas categorias: predição obrigatória em nós-folha e predição opcional em nós-folha.

Na primeira categoria, todos os exemplos devem ser associados com classes representadas por nós-folha. Assim, ao predizer uma classe que está em um determinado nível estará também predizendo as classes que estão nos níveis acima, ou seja, as suas classes ancestrais. A Figura 13 ilustra um exemplo deste tipo de predição onde somente as classes denominadas de H, I, E, F, e G devem ser preditas.

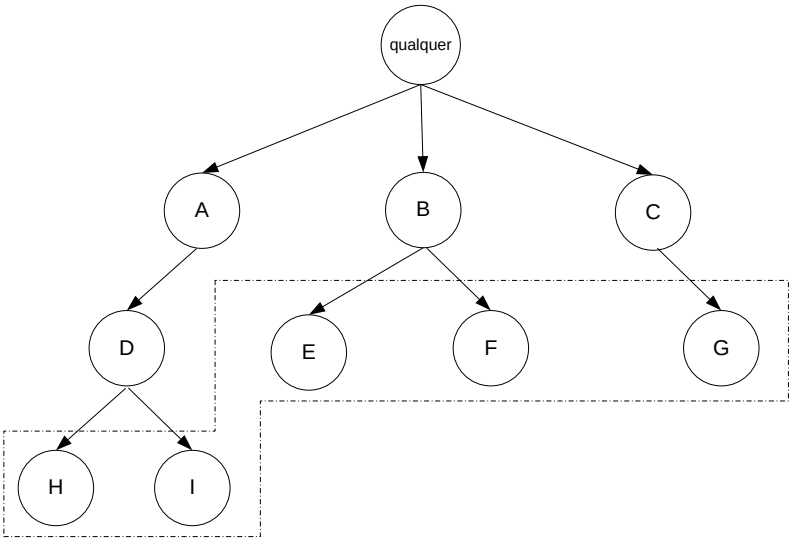


Figura 13 - Exemplo de predição obrigatória em nós-folha.

Já na segunda categoria não existe a obrigatoriedade de que a predição ocorra em nós-folha. Dessa forma, os exemplos podem ser associados a classes que são representadas por qualquer nó interno da hierarquia de classes e seus ancestrais. A Figura 14 ilustra esse tipo de predição onde qualquer nó da hierarquia pode ser predito.

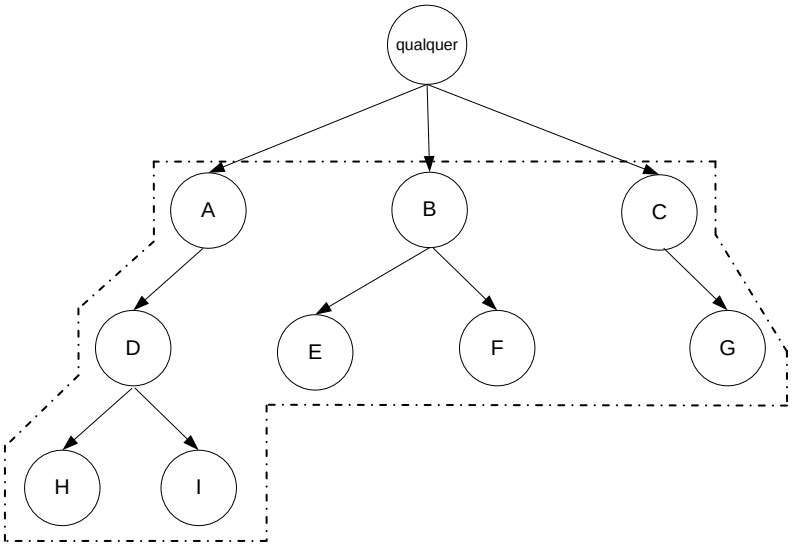


Figura 14 - Exemplo de predição opcional em nós-folha.

3.3 ABORDAGENS PARA TRATAR PROBLEMAS DE CLASSIFICAÇÃO HIERÁRQUICA

Para explorar problemas de classificação hierárquica algumas soluções têm sido propostas, as quais podem ser divididas em três abordagens principais denominadas de classificação hierárquica plana, classificação hierárquica local e classificação hierárquica global (FREITAS & CARVALHO, 2007). Essas abordagens indicam como os classificadores são construídos e não um método de classificação, como o método *top-down* que muitas vezes é citado na literatura como sendo uma das abordagens (SILLA & FREITAS, 2011).

3.3.1 Classificação Hierárquica Plana

A classificação hierárquica plana tem o mesmo comportamento de um algoritmo de classificação convencional na fase de treinamento e teste. Esta abordagem considera que um problema de classificação hierárquica pode ser transformado em um problema de classificação plana desconsiderando o conceito de ancestral e descendente, ou seja, ignora-se

a hierarquia de classe, predizendo apenas os nós-folha. Está técnica é parecida com a classificação plana convencional e pode ser aplicada em estruturas do tipo árvore e do tipo DAG.

Um exemplo desse tipo de predição pode ser observado na Figura 15. Utilizando o exemplo de hierarquia mostrado nesta figura pode-se definir que as classes a serem previstas são: D, H, I, F e G.

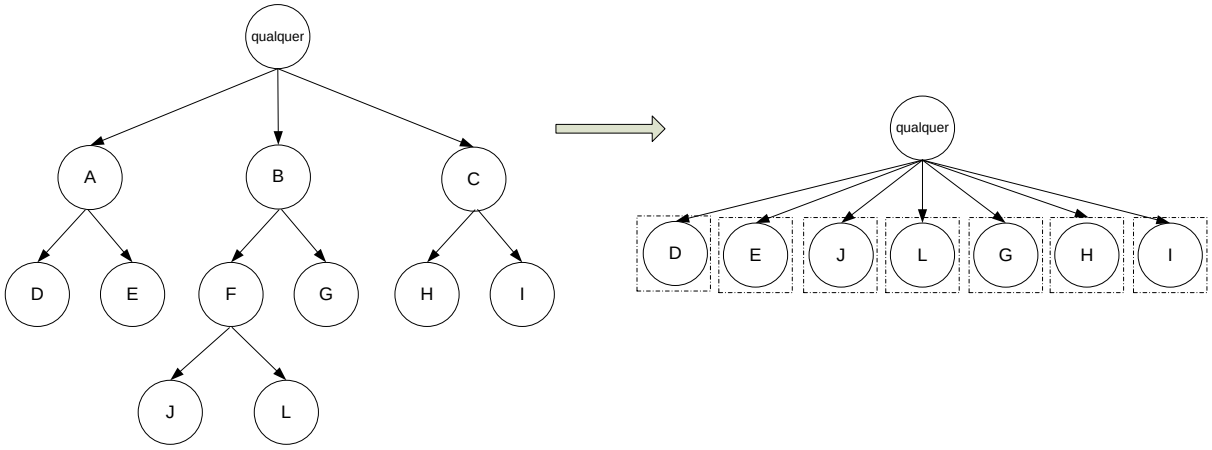


Figura 15 - Exemplo de classificação plana multirrótulo.

3.3.2 Classificação Hierárquica Local

Conforme Silla & Freitas (2010) essa abordagem pode ser divida em três técnicas de classificação. São elas: classificação hierárquica local para cada nó, classificação hierárquica local em nós pais e classificação hierárquica local por nível.

3.3.2.1 Classificação Hierárquica Local para cada nó

Essa é uma das técnicas mais usada na literatura (Xue et al., 2008), (Valentini, 2009), (Barutcuoglu et al., 2006), (Guan et al., 2008), (Jin et. al., 2008). Ela consiste em treinar um classificador binário para cada nó da hierarquia de classes (Figura 16), ou seja, consiste em usar M classificadores locais independentes, um para cada classe (M é o número total de nós na hierarquia de classe). Consequentemente, o número de classificadores que devem ser treinados poderá ser enorme em situações onde há muitas classes. Além disso, um problema crucial que essa técnica pode apresentar é que os resultados poderão ser incompatíveis, pois não há nenhuma garantia que a hierarquia de classes será respeitada.

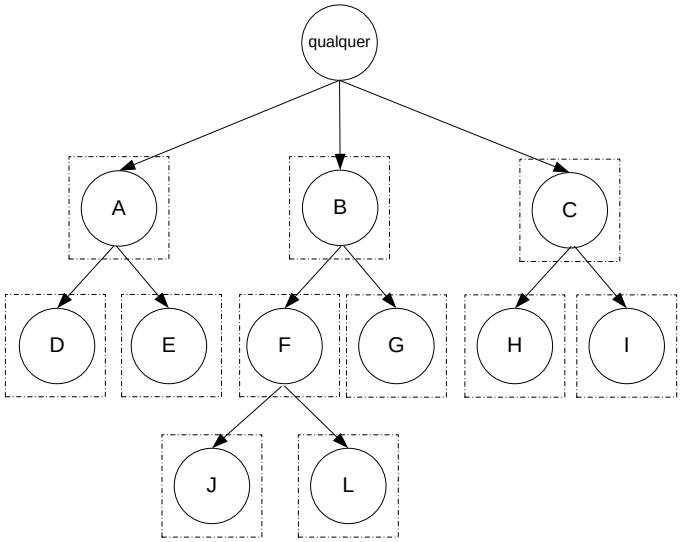


Figura 16 - Exemplo de classificação hierárquica local por nó.

Há várias maneiras de definir o conjunto de amostras positivas e negativas para o classificador binário como citado por Silla & Freitas (2010).

Essa técnica tem a vantagem que cada modelo de classificação é construído utilizando um processo de modularização. Porém, a classificação errada de uma classe em um determinado nível tende a se propagar para os níveis mais profundos da hierarquia, o que representa uma desvantagem para esta abordagem.

3.3.2.2 Classificação Hierárquica Local em nós Pais

Essa técnica consiste em treinar para cada nó pai da hierarquia de classes um classificador multirrótulo. A Figura 17 ilustra um exemplo de classificação local em nós pais. Por exemplo, supondo que no primeiro nível o classificador prediz a classe como sendo B. Na sequência o classificador é treinado com as classes filhas da classe predita anteriormente, no exemplo às classes F e G, evitando o problema de predições inconsistentes e respeitando o relacionamento entre as classes.

Essa técnica não pode ser utilizada em estruturas do tipo DAG sem que haja alguma estratégia para criação dos classificadores pais, visto que um nó filho pode ter mais de um nó pai.

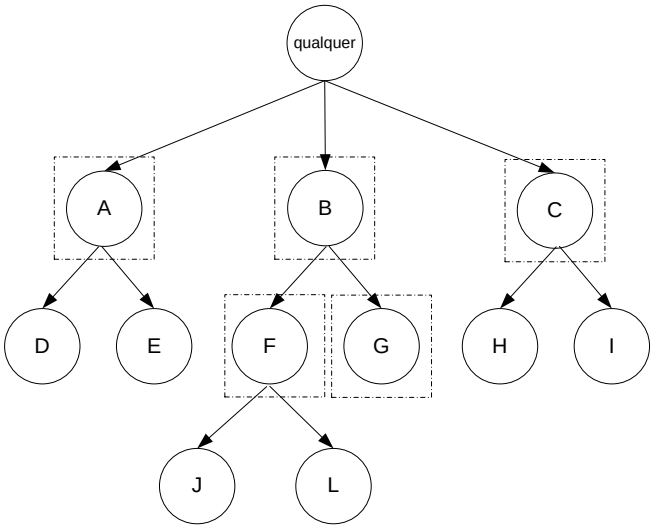


Figura 17 - Exemplo de classificação local por pais.

Uma extensão deste tipo de classificador local é o classificador seletivo proposto por Secker et al. (2007). Ao contrário da classificação local convencional o classificador seletivo permite que mais de uma técnica de aprendizagem de máquina seja utilizada (SECKER et al., 2007). Esse método utiliza um procedimento seletivo para decidir qual técnica será usada em cada iteração da fase de treinamento do classificador. Quando aplicado juntamente com o método *top-down*, ele consiste em dividir o conjunto de treinamento em duas partições, sendo uma delas destinada ao treinamento do classificador para cada técnica de aprendizagem de máquina que está sendo usada e a outra para a validação dos classificadores. Assim, a técnica que gerar o classificador com maior taxa de acerto no conjunto de validação é selecionada para a indução do classificador para o nó em questão. No caso de empate, entre duas ou mais técnicas, deve ser utilizado algum mecanismo de desempate. Após a seleção da técnica, essa é utilizada novamente para o treinamento de um classificador usando todo o conjunto de treinamento. Esse processo é repetido para todos os nós em que o classificador deve ser utilizado. Ao final da fase de treinamento tem-se uma árvore de classificadores de diferentes técnicas de aprendizagem de máquina.

3.3.2.3 Classificação Hierárquica Local por Nível

Esta técnica consiste em criar um classificador multirrótulo para cada nível da hierarquia. A Figura 18 ilustra um exemplo de classificação usando essa técnica. Observa-se que seriam criados três classificadores, um para cada nível da hierarquia.

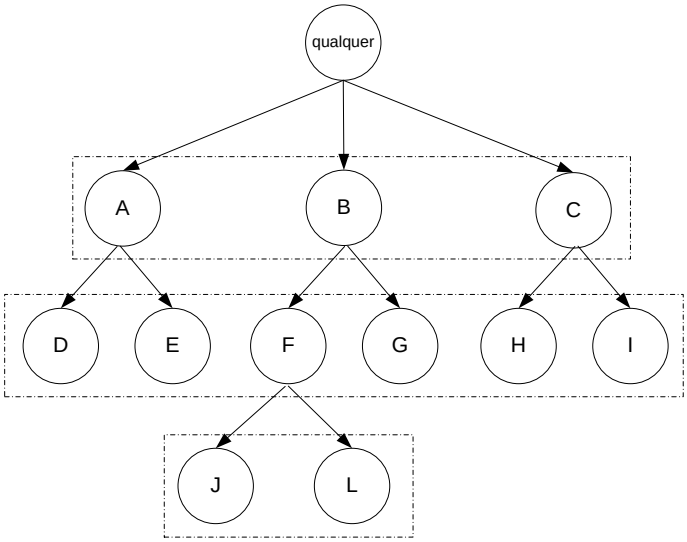


Figura 18 - Exemplo de classificação local por nível.

Esta técnica pode ser usada em estruturas em árvore e em DAG. Em um DAG a aplicação desta técnica é mais complexa, visto que pode existir mais de um caminho nesse tipo de estrutura. Assim, uma classe pode pertencer a mais de um nível na hierarquia o que pode trazer redundância entre os classificadores.

Essa técnica também tem a vantagem que cada modelo de classificação é construído utilizando um processo de modularização como na técnica de classificação hierárquica local para cada nó. O mesmo problema pode ocorrer caso um nó classe tenha sido classificado errado propagando-se para os níveis abaixo da hierarquia.

Em geral, para a avaliação dos classificadores hierárquicos locais pode ser usado o método *top-down* como já comentado anteriormente. Assim, primeiramente, uma amostra é classificada como pertencente a uma ou mais classes no primeiro nível da hierarquia. Na sequência, essa amostra é submetida ao(s) classificador(es) do segundo nível, que são os filhos das classes previstas no primeiro nível, para predizer a qual das subclasses ela pertence. Esse processo é repetido para as classes de nível mais profundo (FREITAS & CARVALHO, 2007).

No trabalho de Xue et al (2008), em que os autores trabalhavam com texto, foi proposta uma estratégia de poda da hierarquia. Assim, quando um documento novo vai ser classificado ele pode ser relacionado com algumas classes da hierarquia. Esse método computa a semelhança entre o documento novo e todos os outros documentos, e realiza a poda da hierarquia. Apenas as classes que permaneceram na hierarquia serão usadas na

próxima fase para a classificação do documento usando o método *top-down* (SILLA & FREITAS, 2010).

3.3.3 Classificação Global ou *Big-Bang*

A abordagem de classificação hierárquica global ou *Big-Bang*, que é o foco deste trabalho, consiste em construir um único modelo de classificação levando em consideração a hierarquia de classes de todo o conjunto de treinamento, conforme pode ser visualizado na Figura 19. Nessa abordagem, a predição pode ocorrer em qualquer nível da hierarquia. Dessa maneira, nenhuma das técnicas utilizadas na classificação plana poderá ser utilizada, sem que sejam feitas alterações no classificador.

As vantagens principais dessa abordagem são que não há necessidade de treinar um número grande de classificadores e a manipulação automática da inconsistência de predição de classe. Sua desvantagem principal é a maior complexidade do classificador global.

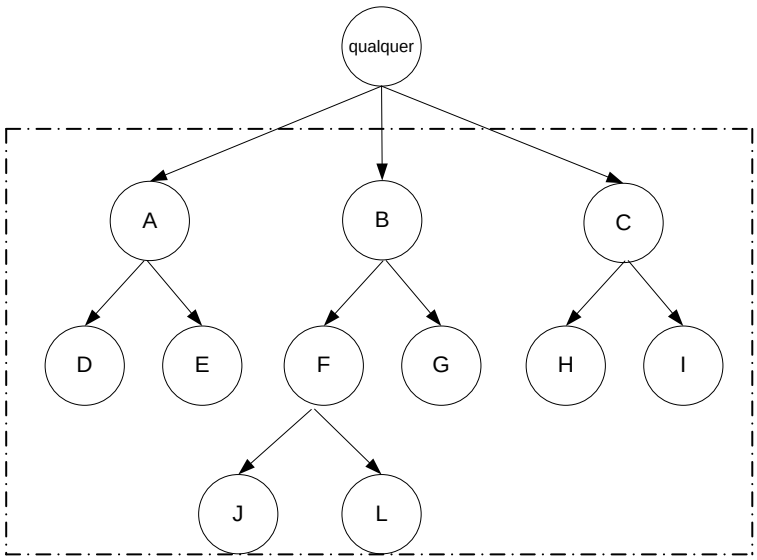


Figura 19 - Exemplo de classificação global.

3.4 AVALIAÇÃO DOS CLASSIFICADORES HIERÁRQUICOS

Sabe-se que as medidas de avaliação de desempenho convencionais foram desenvolvidas para modelos de classificação onde não existe um relacionamento hierárquico entre as classes.

Para os problemas de classificação hierárquica essas medidas precisam ser adaptadas para o tipo de avaliação que se deseja fazer. Não existe um consenso geral para avaliar modelos hierárquicos.

Uma questão que deve ser levada em consideração antes de avaliar qualquer modelo é a maneira como será obtido o resultado, podendo ser obtido uma taxa de desempenho para todo o modelo ou uma taxa de desempenho para cada nível ou para cada classe.

Algumas medidas para avaliar o desempenho de modelos hierárquicos foram propostas, as quais podem ser agrupadas em categorias tais como: medidas baseadas em distância, medidas baseada em semântica, medidas baseada em matriz de custo, medidas baseadas na relação de ancestralidade e descendência entre outras.

3.4.1 Medidas Baseada em Distância

Essa medida de desempenho consiste em atribuir um custo que é proporcional à distância entre o exemplo da classe predita e a classe verdadeira. Essa distância é definida como o número de ligações entre os nós pertencentes ao menor caminho que liga a classe verdadeira e a classe predita. Essa categoria pode ser dividida em duas subcategorias: medida independente da profundidade e medida dependente da profundidade.

3.4.1.1 Medida Independente da Profundidade

Nessa categoria, a medida de distância é calculada sem levar em consideração o nível das duas classes na hierarquia. A distância entre dois nós na hierarquia é definida como o número de extremidades no caminho mais curto que os conecta.

Na estrutura hierárquica em árvore o cálculo da distância é simples. Um exemplo dessa medida é mostrado na Figura 20. Observa-se que a distância entre classes M e G na hierarquia nessa estrutura é igual a três.

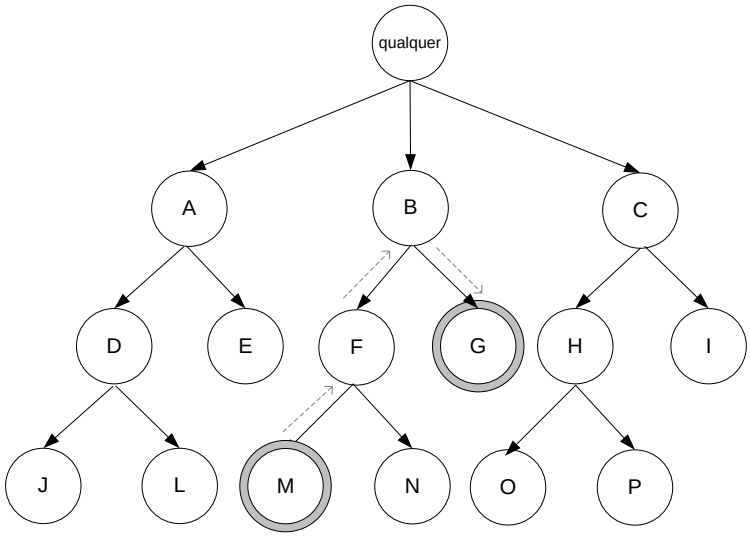


Figura 20 - Exemplo Medida de Distância Independente de Profundidade em Árvores.

A situação é mais complexa em um DAG onde pode haver múltiplos caminhos entre um par de nós, ou seja, entre o nó verdadeiro e o nó predito. Por exemplo, há dois caminhos entre o nó de classe O e o nó F, sendo mostrados na Figura 21.

Neste caso, faz-se necessário a escolha de algum critério para selecionar o caminho entre o nó de classe verdadeira e o nó de classe predito. Um critério que pode ser usado é o do menor caminho. Nesse exemplo, segundo este critério o exemplo selecionado seria o da Figura 21b cuja distância é igual a dois.

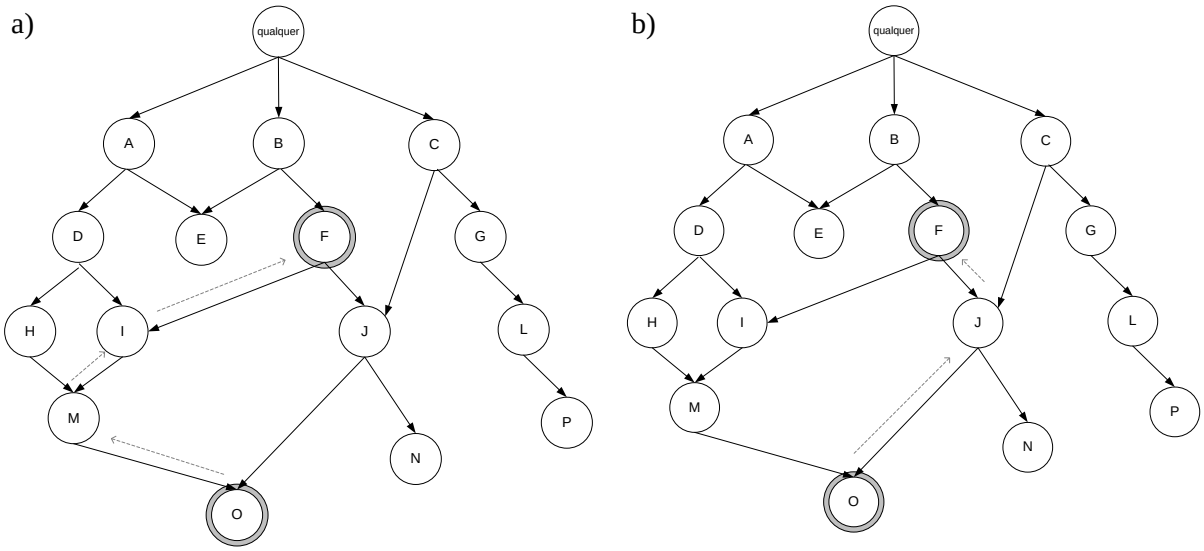


Figura 21 - Exemplo Medida de Distância Independente de Profundidade em DAG.

3.4.1.2 Medida Dependente da Profundidade

Essa medida tem como objetivo considerar o nível de hierarquia no cálculo da distância entre as duas classes. Para isso, considera-se o número de ligações entre as duas classes e a profundidade das classes na hierarquia.

Essa medida pode ser aplicada em problemas de predição obrigatória em nós folha ou em problemas de predição opcional em nós folha. Além disso, as penalizações para os erros de predição nos níveis superiores tende a ser maiores que nos níveis mais profundos.

Por exemplo, suponha um exemplo de uma estrutura em árvore que tem 4 níveis, conforme a Figura 22. A distância entre o nó folha 1.1.1.1 e a classe predita 1.1.1.2 é igual a dois como pode ser observado na Figura 22a. Nota-se que ambos os nós são irmãos, filhos do nó 1.1.1 e estão separados por uma extremidade. Suponha-se a predição em um nó interno da hierarquia no segundo nível (problema de predição de nó folha opcional), por exemplo, o nó 1.2 cuja classe verdadeira é 1.1 como pode ser observado na Figura 22b. A distância entre a classe verdadeira e a classe predita também é igual a dois. Apesar de ambas as distâncias calculadas apresentarem o mesmo valor, um erro na predição do segundo nível (mais geral) é considerado mais grave que o erro na predição do quarto nível (mais específico).

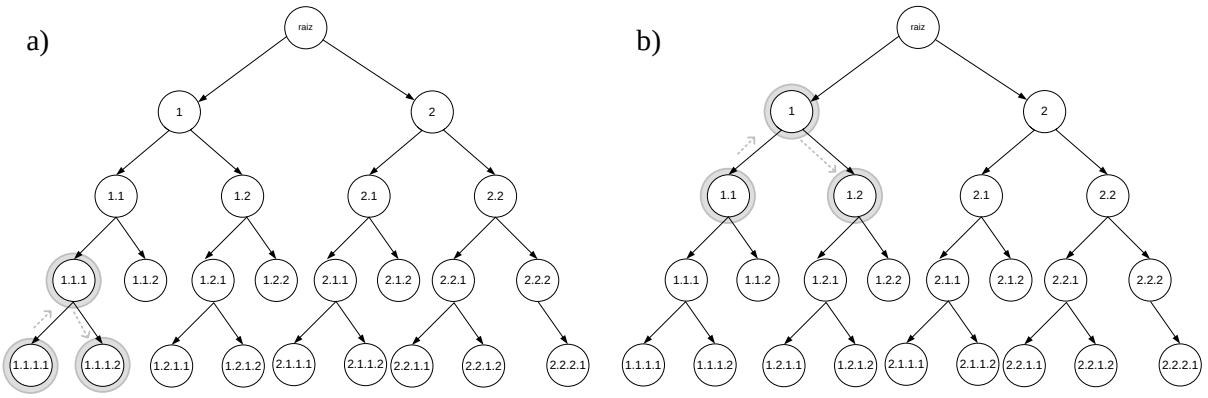


Figura 22 - Exemplo de Predição de Classe em Estrutura de Árvore usando a Medida de Distância Dependente de Profundidade.

Para tratar essa diferenciação entre os níveis pode-se atribuir custos para cada nível de predição na hierarquia de classe. O critério proposto Blockeel et al (2002) foi atribuir pesos nas ligações. As ligações que estão em um nível mais profundo recebem um peso menor das que estão em níveis menos profundos. Assim, ao computar a distância de um caminho da hierarquia utilizam-se os pesos das ligações e calcula-se a distância ponderada entre as classes. Um exemplo pode ser visualizado na Figura 23 que mostra a atribuição de pesos para

as ligações. Observa-se que os pesos no quarto nível são menores que os pesos do primeiro nível da hierarquia.

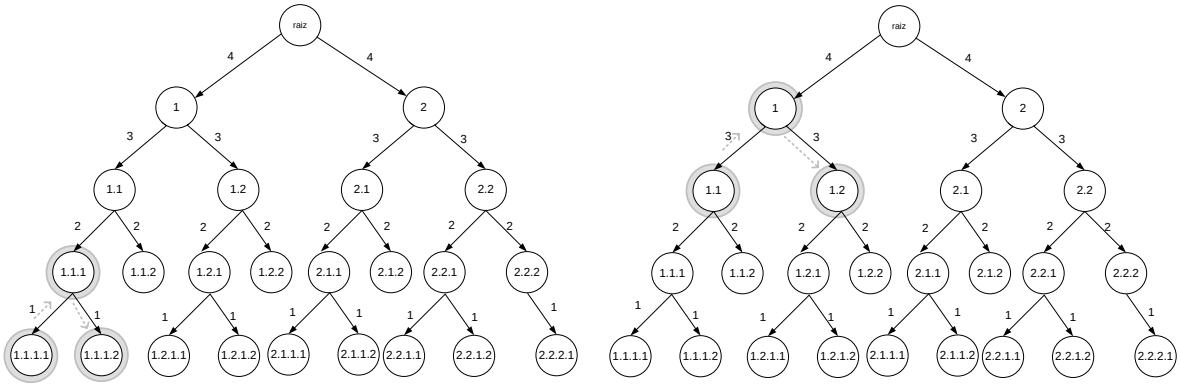


Figura 23 - Exemplo de Predição de Classe em Estrutura de Arvore usando a Medida de Distância Dependente de Profundidade com Atribuição de Pesos.

Com isso, o erro da predição para as classes que estão em um nível mais alto da hierarquia (mais próximas da raiz) será considerado mais grave do que as que estão em um nível mais profundo.

Para a estrutura do tipo DAG, onde pode haver múltiplos caminhos, devem-se seguir os mesmos passos usados na medida independente de profundidade, porém incluindo o peso das ligações das classes.

Uma maneira de se obter a taxa de acerto (TA), taxa de erro (TE), sensibilidade (S) e precisão (P) para a classificação hierárquica, proposta por Sun & Lim (2001), é considerar os erros de predição baseada na distância. Para isso é calculada a contribuição de falsos positivo F_pCon_i para cada classe C_i da hierarquia através da Equação 16.

$$FpCon_i = \sum_{d \in FP_i} RCon(d, C_p) \qquad \textbf{Equação 16}$$

em que d é um exemplo de entrada, C_p a classe predita, C_v a classe verdadeira e $RCon$ é a contribuição refinada, que serve para normalizar a contribuição de cada exemplo no intervalo de $[-1,1]$. Essa é obtida pela Equação 17.

$$RCon(d, C_p) = \min(1, \max(-1, Con(d, C_v))) \qquad \textbf{Equação 17}$$

Para que seja possível calcular a $RCon$ faz-se necessário calcular a contribuição de falsos positivos Con para cada classe. Esse cálculo é obtido pela fórmula mostrada na Equação 18.

$$Con(d, C_p) = 1.0 - \frac{Dis(C_p, C_v)}{Dis_\theta}$$ **Equação 18**

em que $Dis(C_p, C_v)$ é a distancia entre a classe predita C_p e a classe verdadeira C_v, e Dis_θ é a distância aceitável, especificada pelo usuário e devendo ser maior do que zero.

Já a contribuição de falsos negativos $FnCon_i$ é obtida de maneira semelhante como pode ser observado na Equação 19, Equação 20 e Equação 21.

$$FnCon_i = \sum_{d \in FN_i} RCon(d, C_v)$$ **Equação 19**

$$RCon(d, C_v) = \min(1, \max(-1, Con(d, C_p)))$$ **Equação 20**

$$Con(d, C_v) = 1.0 - \frac{Dis(C_p, C_v)}{Dis_\theta}$$ **Equação 21**

Calculando o valor de $FpCon_i$ e $FnCon_i$ a TA é obtida na Equação 22, a TE pela Equação 23, a P na Equação 24 e a S Equação 25 (SUN & LIM, 2001).

$$TA = \frac{|VN| + |VP| + FpCon_i + FnCon_i}{|VN| + |VP| + |FP| + |FN|}$$ **Equação 22**

$$TE = \frac{|FN| + |FP| - FpCon_i - FnCon_i}{|VN| + |VP| + |FP| + |FN|}$$ **Equação 23**

$$S = \frac{\max(0, |TP_i| + FpCon_i + FnCon_i)}{|VP_i| + |FN_i| + FpCon_i}$$ **Equação 24**

$$P = \frac{\max(0, |VP_i| + FpCon_i + FnCon_i)}{|VP_i| + |FP_i| + FnCon_i}$$ **Equação 25**

Outra maneira de utilizar a distância no cálculo da TE consiste em calcular o erro de predição de cada exemplo predito. A TE é obtida através da média do erro medido para todas as amostras. Para cada exemplo classificado, é calculada a distância entre a classe predita e a classe verdadeira. Na sequência essa distância é normalizada $Dist_N$ como mostra a Equação 26, em que C_d é a classe mais distante da classe verdadeira.

$$Dist_N = \frac{Dis(C_p, C_v)}{Dis(C_d, C_v)}$$ **Equação 26**

Após calcula-se a TE que é obtida pelo somatório da distância normalizada de todos os exemplos que foram preditos (os exemplos que foram preditos corretamente terão o valor de distância igual a zero) dividido pela quantidade total de exemplos preditos $NTotal$ (Equação 27).

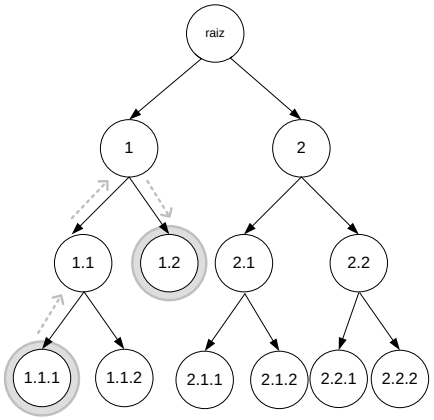
$$TE = \sum_{i=1}^{NTotal} Dist_N i$$ **Equação 27**

Obtida a TE a TA pode ser então deduzida como mostra a Equação 11.

3.4.2 Medida Baseada em Matriz de Custo

Essa categoria consiste em especificar o custo para cada erro de classificação usando uma matriz de custo (FREITAS & CARVALHO, 2007). Essa matriz é uma generalização da matriz de custos de erro de classificação para problemas de classificação plana (WITTEN & FRANK, 2005).

Há diferentes maneiras de fazer essa generalização. Uma delas consiste em determinar o custo associado a cada erro de classificação. Por exemplo, para um problema de predição obrigatória em nó folha uma matriz de custos em uma estrutura hierárquica com dois níveis é mostrada na Figura 24.



				Classe Verdadeira						
				1			2			
				1.1		1.2	2.1		2.2	
				1.1.1	1.1.2	1.2	2.1.1	2.1.2	2.2.1	2.2.2
Classe Predita	1	1.1	1.1.1	0	a	**b**	e	e	E	E
			1.1.2	A	0	b	e	e	E	E
		1.2	1.2	B	b	0	d	d	D	D
	2	2.1	2.1.1	E	e	d	0	a	C	C
			2.1.2	E	e	d	a	0	C	C
		2.2	2.2.1	E	e	d	c	c	0	A
			2.2.2	E	e	d	c	c	A	0

Figura 24 - Exemplo de Predição Hierárquica usando Medida de Avaliação baseada na Matriz de Custo.

Observa-se que, nas células da diagonal principal da tabela foi atribuído o valor zero. Isso significa que não houve erro de classificação. Para as outras células o custo variou de $a,...,e$ e que os valores de cada um desses custos são maiores que zero.

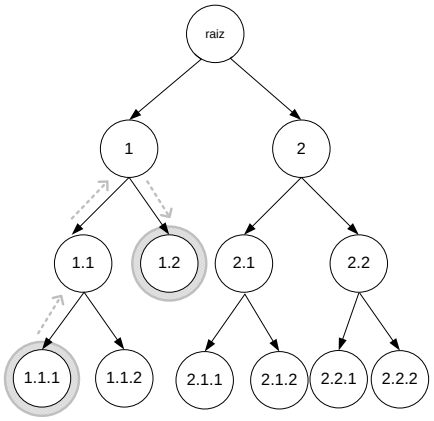
Assim, ao se fazer uma predição errada como, por exemplo, para a classe 1.2 cuja classe verdadeira desse exemplo é 1.1.1 atribui-se um custo para essa predição, nesse caso o custo atribuído foi b.

Conforme Freitas & Carvalho (2007) pode-se utilizar essa medida juntamente com as medidas descritas anteriormente.

Para obter o erro de classificação em uma estrutura de árvore baseada na distância, onde o custo é obtido conforme o número de ligações, basta atribuir um custo para os nós irmãos de uma subárvore e outro custo para os nós primos desta subárvore.

Seguindo o mesmo exemplo de predição descrito anteriormente, o custo atribuído para as classes 1.1.1 e 1.1.2 (que são nós irmãos) é igual a dois conforme pode ser observado na Figura 25.

No caso de predição opcional em nó folha basta estender a matriz de custo para as classes internas da hierarquia (FREITAS & CARVALHO, 2007).



				Classe Verdadeira						
				1			2			
				1.1		1.2	2.1		2.2	
				1.1.1	1.1.2	1.2	2.1.1	2.1.2	2.2.1	2.2.2
Classe Predita	1	1.1	1.1.1	0	2	**3**	6	6	6	6
			1.1.2	2	0	3	6	6	6	6
		1.2	1.2	3	3	0	5	5	5	5
	2	2.1	2.1.1	6	6	5	0	2	4	4
			2.1.2	6	6	5	2	0	4	4
		2.2	2.2.1	6	6	5	4	4	0	2
			2.2.2	6	6	5	4	4	2	0

Figura 25 - Exemplo de Predição Hierárquica usando Medida de Avaliação baseada na Matriz de Custo juntamente com a medida baseada em Distância.

3.4.3 Medidas Baseada na Relação de Ancestralidade e Descendência

Um exemplo dessa medida foi utilizado por Ipeirotis et al (2001). Neste trabalho, foi utilizado o conceito de classes descendentes, considerando as subárvores enraizadas na classe predita e na classe verdadeira. Cada subárvore é formada pela classe em si e pelas classes descendentes, ou seja, as suas subclasses.

Segundo Ipeirotis et al (2001) as medidas de precisão e sensibilidade da hierarquia podem ser calculadas através da intersecção dessas subárvores.

A medida de Precisão Hierárquica Ph, como mostrada na Equação 28, é obtida pela divisão do número de descendentes comuns pelo número de descendentes da classe predita.

A medida de Sensibilidade Hierárquica Sh, como pode ser observada na Equação 29, é obtida pela divisão do número de descendentes comuns pelo número de descendentes da classe verdadeira.

$$Ph = \frac{\left|Desc(C_v) \cap Desc(C_p)\right|}{\left|Desc(C_p)\right|}$$ **Equação 28**

$$Sh = \frac{\left|Desc(C_v) \cap Desc(C_p)\right|}{\left|Desc(C_v)\right|}$$ **Equação 29**

em que $Desc(C_v)$ e $Desc(C_p)$ representa as classes contidas na subárvore em que C é a classe raiz (incluindo C) da classe verdadeira e da classe predita, respectivamente. Essas medidas são então utilizadas no cálculo de uma extensão hierárquica da Medida-F como mostra a Equação 30.

$$Medida - Fh = \frac{(\beta^2 + 1) * Ph * Sh}{1 * Ph + Sh}$$ **Equação 30**

em que β geralmente recebe o valor igual a 1.

O problema dessa medida é que ela assume que a classe predita é uma classe descendente ou uma classe ancestral da classe verdadeira. Quando essas classes estão no mesmo nível, a intersecção é um conjunto vazio.

Outro exemplo de medida pertencente a essa categoria pode ser encontrado em Kiritchenko et al (2004) onde é usado o conceito de ancestralidade. Assim, a medida de precisão (Equação 31) é obtida pela divisão do número de ancestrais comuns pelo número de ancestrais da classe predita. Já a medida de sensibilidade (Equação 32) é obtida pela divisão do número de ancestrais comuns pelo número de ancestrais da classe verdadeira. Em ambas as medidas, o nó raiz não é considerado ancestral da classe C.

$$Ph = \frac{\left|Anc(C_v) \cap Anc(C_p)\right|}{\left|Anc(C_p)\right|}$$ **Equação 31**

$$Sh = \frac{\left|Anc(C_v) \cap Anc(C_p)\right|}{\left|Anc(C_v)\right|}$$ **Equação 32**

em que $Anc(C_v)$ e $Anc(C_p)$ são as classes ancestrais da classe verdadeira e predita, incluindo a própria classe C, respectivamente.

3.4.4 Medida Baseada na Curva de Precisão e Revocação

Vens et. al. (2008) propôs uma medida baseada na análise de curvas de precisão e revocação (curvas PR). Tal medida funciona da seguinte maneira: um conjunto de limiares entre 0 e 1 são selecionados. Cada limiar corresponde a um ponto no espaço da curva PR e variando esses limiares obtêm-se a curva PR. Entretanto, para um cálculo mais real possível da área da curva, o uso de algum método de interpolação deve ser utilizado, como por exemplo, o método de interpolação não-linear (DAVIS & GOADRICH, 2006).

Para um determinado limiar um ponto de Precisão e Revocação $(\overline{\Pr ec}, \overline{\mathrm{Re} v})$ no espaço da curva PR são obtidos através da Equação 33 e da Equação 34, respectivamente, em que i representa as classes.

$$\overline{\Pr ec} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FP_i} \qquad \textbf{Equação 33}$$

$$\overline{\mathrm{Re} v} = \frac{\sum_i VP_i}{\sum_i VP_i + \sum_i FN_i} \qquad \textbf{Equação 34}$$

Essa medida funciona apenas para classificadores que possuem em sua saída um valor contínuo. Dessa maneira, avalia-se a saída desse classificador e compara com um determinado limiar. Assim, se uma saída é maior que o limiar então se atribui a instância à classe, caso contrario a instância não é atribuída a classe.

Quanto menor for o valor do limiar mais exemplos são atribuídos a uma classe aumentando a medida de revocação.

Outra questão levantada pelos autores é que as curvas PR podem ser construídas individualmente para cada classe em um problema multirrótulo, obtendo como positivos os exemplos pertencente à classe e como negativo os outros exemplos. Para combinar os desempenhos individuais em cada classe de forma a obter o desempenho geral, dois métodos podem ser utilizados: a área abaixo da curva PR $(AUPRC)$ e a área média abaixo da curva PR (Vens et al., 2008).

A área média abaixo da curva é descrita por $AU(\overline{PRC})$. Nessa medida é calculada a média ponderada das áreas abaixo da curva PR individuais. A Equação 35 mostra como é calculada essa abordagem.

$$\overline{AUPPC_{w1,\ldots,w|C|}} = \sum_i w_i * AUPRC_i \qquad \textbf{Equação 35}$$

Os autores utilizaram essa abordagem de duas maneiras. A primeira, denominada de $\overline{AUPRC}$, os pesos w_i são inicializados com o valor de $1/|C|$, em que C representa todas as classes. A segunda abordagem, denominada de $\overline{AUPRC}_w$, consiste em atribuir um peso para uma classe conforme a sua frequência. Essa frequência é calculada por $w_i = v_i / \sum_j v_j$, em que v_i é a frequência de classe c_i do conjunto de dados. Nesse caso, segundo os autores é que em alguns problemas de classificação hierárquica, classes que são mais frequentes podem ser mais importantes.

3.5 TESTES ESTATÍSTICOS PARA VALIDAÇÃO DOS RESULTADOS

A análise estatística dos resultados obtidos de um determinado estudo é uma ferramenta muito importante na validação desses dados.

Os testes estatísticos são fundamentalmente utilizados em pesquisas que tem como objetivo comparar condições experimentais. Existe uma série de testes estatísticos que podem auxiliar as pesquisas. Os testes estatísticos fornecem um respaldo científico às pesquisas para que estas tenham validade e tenham aceitabilidade no meio científico.

Os testes podem ser divididos em paramétricos e não-paramétricos. A diferença entre ambos os testes refere-se ao tipo de valores da variável estudada. Nos testes paramétricos os valores da variável devem ter distribuição normal ou aproximação normal. Já os testes não-paramétricos não têm exigências quanto ao conhecimento da distribuição da variável na população.

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

3.5.1 Teste de Wilcoxon

O teste de Wilcoxon (1945) é aplicado quando estão em comparação dois grupos relacionados e a variável deve ser de mensuração ordinal. O teste classifica em postos a diferença entre os algoritmos sobre cada base usada para avaliação de desempenho. Em seguida, soma as diferenças positivas (Equação 36) e as negativas (Equação 37).

$$W^+ = \sum_{d_i>0} r_i$$ **Equação 36**

$$W^- = \sum_{d_i<0} r_i$$ **Equação 37**

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

Na sequência calcula-se o valor T que representa a menor das somas de postos de mesmo sinal (Equação 38) (DESMAR, 2006).

$$T = \min(W^+; W^-)$$ **Equação 38**

Em seguida determina-se o valor de N que é o total das diferenças com sinal. Se $N \leq 25$, os valores críticos de T são tabelados, onde N representa o número de base de dados avaliadas, descontados o número de empates $(d_i = 0)$. Já para os valores em que $N > 25$ utiliza-se a estatística z (Equação 39) pois se considera que a distribuição dos dados é aproximadamente normal.

$$z = \frac{T - \frac{1}{4}n(n+1)}{\sqrt{\frac{1}{24}n(n+1)(2n+1)}}$$ **Equação 39**

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

3.5.2 Teste de Friedman

O teste de Friedman é um teste não paramétrico (FRIEDMAN, 1937, 1940) recomendado para comparar o desempenho de vários algoritmos sobre diferentes bases de dados (DESMAR, 2006).

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

Após o calculo da média dos postos médios dos classificadores é necessário calcular a diferença estatística existente entre eles. Este calculo é feito através da aplicação da Equação 40.

$$X_F^2 = \frac{12n}{k(k+1)}\left[\sum_j r_j^2 - \frac{k(k+1)^2}{4}\right]$$ **Equação 40**

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

Segundo Iman e Davenport (1980) apud (DESMAR, 2006) o teste de Friedman é considerado muito conservador, e por isso, sugere a utilização da estatística F_F mostrada na Equação 41 distribuída de acordo com a distribuição *F* de Snedcor com $k-1$ e $(k-1)*(n-1)$ graus de liberdade.

$$F_F = \frac{(n-1)X_F^2}{n(k-1) - X_F^2}$$ **Equação 41**

Se a hipótese nula é rejeitada Demsar (2006) sugere o teste de Nemenyi, que calcula a distância crítica (CD - *critical distance*) entre desempenhos através da Equação 42.

$$CD = q_a\sqrt{\frac{k(k+1)}{6N}}$$ **Equação 42**

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

4 TRABALHOS CORRELATOS

Este Capítulo trata sobre os trabalhos relacionados à classificação hierárquica. Como o objetivo do trabalho é o desenvolvimento de um algoritmo global para a classificação hierárquica de estruturas do tipo DAG, dar-se-á mais detalhes aos trabalhos que estão diretamente relacionados a este tema.

Na Seção 4.1 mostra-se um panorama dos trabalhos desenvolvidos que utilizam a classificação hierárquica. Na Seção 4.2 são descritos os principais trabalhos desenvolvidos para estruturas hierárquicas do tipo DAG. A Seção 4.3 cita os trabalhos cujo classificador utilizado é uma rede neural artificial e estratégia evolucionária e por fim, na Seção 4.4 são feitas algumas considerações sobre os trabalhos citados.

4.1 PANORAMA DOS TRABALHOS QUE UTILIZAM A CLASSIFICAÇÃO HIERÁRQUICA

A classificação hierárquica é uma técnica que vem sendo bastante utilizada na área de mineração de textos desde a década de 90. Entre os trabalhos desenvolvidos neste contexto pode-se citar Koller & Sahami (1997), Sun & Lim (2001), Sun et al (2003), Kiritchenko et al (2005), Kiritchenko et al (2006).

Entre as motivações para as pesquisas nesta área está o grande número de documentos eletrônicos, os quais, virtualmente, podem ser organizados hierarquicamente, como por exemplo, as páginas web, bibliotecas digitais, correio eletrônico, entre outros (CHAKRABARTI et al, 1998) com o objetivo de facilitar a busca e a recuperação de informações.

Outra área em que a classificação hierárquica vem sendo utilizada é na organização e recuperação de gêneros musicais. Alguns dos trabalhos que usaram hierarquias de classe neste domínio de aplicação são: Burred & Lerch (2003), Barbedo & Lopes (2007), DeCoro et al (2007), Silla & Freitas (2009).

Existem alguns trabalhos de classificação hierárquica sendo aplicados em imagens, como, por exemplo, o trabalho desenvolvido por Dimitrovski et al (2008), que utilizou o método de classificação hierárquica desenvolvido por Vens et al (2008) para a classificação de imagens médicas.

Já na bioinformática cuja área apresenta vários problemas a serem resolvidos por meio da classificação hierárquica, como por exemplo, a predição de função de proteínas, é ainda, relativamente, pouco explorada. Alguns trabalhos têm sido publicados utilizando esta técnica tais como: Clare & King (2003), Vens et al (2008), Silla & Freitas (2009a) entre outros.

Como o foco desta tese não é o desenvolvimento e a aplicação de algoritmo, especificamente, para estrutura hierárquica do tipo árvore e sim para estrutura, mais complexa, do tipo DAG, apenas é citado, na Tabela 4, alguns trabalhos aplicados em estrutura hierárquica do tipo árvore, citando a abordagem utilizada, o algoritmo base e a aplicação.

Tabela 4 - Trabalhos desenvolvidos e aplicados em estruturas do tipo árvore.

Trabalhos	Abordagem		Algoritmo Base	Aplicação
Barbedo & Lopes (2007)	Classificação Plana		Baseado em Vetores *summary features*	Classificação de Gênero Musical
D' Alessio et al (2000)	Classificação Local	Por Nós	Classificador Heurístico Linear	Categorização de Texto
Dumais & Chen (2000)			SVM	Categorização de Texto
Sun & Lim (2001)			SVM	Categorização de Texto
Sun et al (2003)			SVM e Naïve Bayes	Categorização de Texto
Liu et al (2005)			SVM	Categorização de Texto
Cesa-Bianchi et al (2006a,b)			Bayes e SVM	Categorização de Texto
Cesa-Bianchi & Valentini (2009)			Ensembles Bayesianas	Predição de Função de Proteína
Esuli et al (2008)			AdaBoost	Categorização de Texto
Xue et al (2008)			SVM	Categorização de Texto
Bennett & Nguyen (2009)			SVM	Categorização de Texto
Binder et al (2009)			SVM	Classificação de Imagem
Valentini (2009)			Combinação de Classificadores	Predição de Função de Proteína
Valentini & Re (2009)			SVM	Predição de Função de Proteína
Koller & Sahami (1997)		Por Nós Pais	Classificadores Bayesianos	Categorização de Texto
Chakrabarti et al (1998)			Naive Bayes	Categorização de Texto
McCallum et al (1998)			Naive Bayes	Categorização de Texto
Weigend et al (1999)			Rede Neural	Categorização de Texto
Ruiz & Srinivasan (2002)			Rede Neural MLP	Categorização de Texto
Burred & Lerch (2003)			3-component Gaussian Mixture Model	Classificação de Gênero Musical
McKay & Fujinaga (2004)			Rede Neural (para frente) e KNN	Classificação de Gênero Musical
Li & Ogihara (2005)			SVM	Classificação de Gênero Musical
Brecheisen et al (2006)			SVM	Classificação de Gênero Musical
Holden & Freitas (2005, 2006, 2008, 2009)			PSO/ACO	Predição de Função de Proteína

\Xiao et al (2007)			Rede Neural MLP	Classificação de Discurso Emocional
Secker et al (2007)			Selective Top-Down utilizando várias técnicas de AM	Predição de Função de Proteína
Costa (2008)			Árvore de Decisão, Ripper, SVM, K-NN e Rede Bayesiana	Predição de Função de Proteína
Silla & Freitas (2009b)			Naïve Bayes	Classificação de Gênero Musical
Gauch et al (2009)			Rocchio, *k*-NN, SVM e Naïve Bayes	Categorização de Texto
Jensen et al. (2002)		Por Nível	Rede Neural	Predição de Função de Proteína
Clare & King (2003)			Árvore de Decisão	Predição de Função de Proteína
Weinert & Lopes (2004)			Rede Neural	Predição de Função de Proteína
Cerri & Carvalho (2010)			Rede Neural MLP	Predição de Função de Proteína
Labrou & Finin (1999)	Classificador Global		Rocchio	Categorização de Texto
Wang et al (1999)			SVM/Associação de Regras	Categorização de Texto
Clare & King (2003)			Árvore de Decisão	Predição de Função de Proteína
Blockeel et al (2006)			Arvore de Decisão baseada em PCT	Predição de Função de Proteína
Cai & Hofmann (2004, 2007)			SVM	Categorização de Texto
Dekel et al (2004a,b)			SVM	Classificação de Fonema
Peng & Choi (2005)			Naïve Bayes	Categorização de Texto
Rousu et al (2005, 2006)			Classificador baseado em Kernels	Categorização de Texto
Astikainen et al (2008)			Hierarchical Max-Margin Markov algorithm (HM3) and the Maximum Margin Regression algorithm (MMR)	Predição de Função de Proteína
Seeger (2008)			SVM	Categorização de Texto
Silla & Freitas (2009a)			Naive Bayes	Predição de Função de Proteína
Qiu et al (2009)			SVM	Categorização de Texto
Cerri & Carvalho (2011)			Rede Neural MLP	Predição de Função de Proteína

4.2 TRABALHOS ESTRUTURADOS NA FORMA DE DAG

Nesta Seção serão descritos os trabalhos mais relevantes aplicados em estruturas hierárquica do tipo DAG. A Tabela 5 cita os principais trabalhos envolvendo este tipo de hierarquia.

Tabela 5 - Trabalhos desenvolvidos e aplicados em estruturas do tipo DAG.

Trabalhos	Abordagem		Algoritmo Base	Aplicação
Jensen et al. (2003)	Classificação Plana		Rede Neural MLP	Predição de Função de Proteína
Laegreid et al. (2003)			Indução de regras baseada na teoria de Rough Set	Predição de Função de Proteína
Tu et al. (2004)			Rede Neural	Predição de Função de Proteína
Hayete & Bienkowska (2005)			Árvore de Decisão	Predição de Função de Proteína
Barutcuoglu & DeCoro (2006)	Classificação Local	Por Nós	Redes Bayesianas com k-NN	Classificação de Imagens 3D
Barutcuoglu et al (2006)			SVM e Framework Bayesiano	Predição de Função de Proteína
DeCoro et al (2007)			Framework Bayesiano	Classificação de Gênero Musical
Guan et al (2008)			SVM com Rede Bayes	Predição de Função de Proteína
Jin et al (2008)			SVM, Naive Bayes	Predição de Função de Proteína
Kiritchenko et al (2005, 2006)	Classificação Global		AdaBoost	Categorização de Texto
Vens et al (2008)			Árvore de Decisão baseada em PCT	Predição de Função de Proteína
Alves et al (2008)			Sistema Imunológico Artificial e Extração de Regras	Predição de Função de Proteína
Dimitrovski et al (2008)			Arvore de Decisão baseada em PCT	Classificação de Imagem
Aleksovski et al (2009)			Árvore de Decisão baseada em PCT	Predição de Função de Proteína
Otero et al (2009)			Colônia de Formiga	Predição de Função de Proteína
Wang et al (2009)			SVM	Predição de Função de Proteína
Otero et al (2010)			Colônia de Formiga	Predição de Função de Proteína
Bi & Kwok (2011)			Condensing Sort and Select Algorithm (CSSA)	Predição de Função de Proteína

4.2.1 Trabalhos utilizando a Abordagem de Classificação Hierárquica Plana

Jensen et al (2003) utilizaram os dados provenientes do repositório universal de sequências de proteínas InterPro (*Universal Protein Resource*), no qual famílias de proteínas tem sido anotadas seguindo o esquema hierárquico da GO. O conjunto de dados utilizado nos experimentos é formado por 16 atributos, 12501 amostras, as quais foram particionadas em 5 subconjuntos usando o método de validação cruzada e 347 categorias de classes.

Os autores utilizaram a abordagem de classificação plana para a predição hierárquica. Para isso, uma Rede Neural Artificial multicamada, com uma camada de neurônios escondida, foi usada para treinar cada classe.

Várias combinações de atributos foram utilizadas para treinar a rede cuja saída consistia em determinar se a amostra pertencia (valor 1) ou não (valor 0) a classe predita. Da mesma forma, foram feitas combinações de cinco redes neurais, com um único atributo de entrada, para o treinamento dos classificadores. Com isso, a maioria das categorias de classes acabou sendo descartada resultando em 26 classes.

Para cada classe resultante, novas combinações de redes neurais foram treinadas. Primeiramente, fez-se o treinamento utilizando um atributo de entrada e os que eram consideráveis piores foram descartados. Na sequência, fizeram-se combinações de tipos e quantidades de atributos de maneira a obter a melhor combinação de atributos para a arquitetura da rede. Para evitar redundâncias na predição, outra seleção de categorias foi realizada resultando em 14 classes da GO.

Para a avaliação de desempenho do modelo os autores usaram as medidas de sensibilidade e a taxa de falsos positivos.

Nota-se que neste trabalho os autores não levaram em consideração a hierarquia das classes, apesar de estarem utilizando dados que apresentam esta estrutura, pois durante a fase de treinamento apenas a classe atual era tratada como sendo positiva e as demais como negativas ignorando a hierarquia de classes.

O trabalho desenvolvido por Laegreid et al. (2003) também utilizou o esquema de classes do GO. O objetivo do trabalho era construir um modelo capaz de representar o relacionamento entre a expressão dos genes em função do tempo e o seu envolvimento com os processos biológicos, e usar este modelo para predizer a função biológica dos genes não conhecidos.

Para a construção do modelo os autores utilizaram um conjunto de regras do tipo *se-então* usando um algoritmo de indução de regras baseado na teoria de Rough Set e um Algoritmo Genético. O método desenvolvido consiste em obter regras que representem as relações entre expressão gênica em função do tempo e o envolvimento de um gene em processos biológicos. Dessa forma, ao utilizar este modelo de classificação pode-se predizer os papeis biológicos de genes desconhecidos.

O conjunto de dados de treinamento era constituído de informações do perfil de 497 genes. Desses 497 genes, 284 genes eram conhecidos, ou seja, tinham seus termos categorizados na GO, o restante (213 genes) eram desconhecidos.

Dos 284 genes conhecidos 273 deles pertenciam a uma das 23 classes e por isso foram agrupados. Os 273 genes agrupados nas 23 classes deram origem a 549 amostras de

treinamento constituídas por 55 atributos. Isso se deve ao fato de que para 167 genes havia mais de um processo biológico.

Após esta fase de pré-processamento dos dados, o algoritmo de aprendizagem foi aplicado para a geração de um conjunto de regras para cada uma das 23 classes de processos biológicos da GO.

Na fase de treinamento foi utilizada a validação cruzada com 10 partições. O modelo de treinamento foi constituído por 18064 regras. Em média 3 dos 55 atributos foi usado em cada regra. Essa redução no número de atributos deve-se a aplicação do algoritmo genético.

Para 211 genes dos 213 desconhecidos ou não categorizados foram obtidos 548 classificações de possíveis processos biológicos. Como medida de avaliação, os autores usaram a AUC.

Nota-se que neste trabalho os autores também utilizaram a abordagem de classificação plana para a predição hierárquica. Além disso, a hierarquia de classes DAG da GO foi usada apenas na fase de pré-processamento para a construção do conjunto de dados de treinamento.

Observa-se que nos dois trabalhos descritos anteriormente foi desenvolvido um classificador para predizer cada classe da GO. Quando uma nova amostra precisa ser classificada todos os classificadores são testados. O classificador que apresentar melhor medida de confiança em seu resultado é que será utilizado para predizer a classe a qual pertence a amostra (FREITAS & CARVALHO, 2007).

O trabalho desenvolvido por Tu et al. (2004) também utiliza o esquema de hierarquia de classes do GO e a abordagem de predição hierárquica usando um algoritmo de classificação plana assim como nos trabalhos de Jensen et al. (2003) e Laegreid et al. (2003).

Nesse trabalho é proposto um *framework* de predição baseada na capacidade de aprender (*learnability-based predicition*), cujo conceito baseia-se no fato de que nem todas as classes da hierarquia podem ser preditas com a taxa de acerto necessária e que, portanto, deve-se focar naquelas que podem ser preditas com maior confiança.

Assim como no trabalho de Jensen et al. (2003) os autores usaram uma Rede Neural Artificial para a construção do modelo de predição, porém levando em consideração o relacionamento entre as classes ancestrais e descendentes da hierarquia.

Um conjunto com classes ancestrais da GO e suas classes descendentes foi denominado pelos autores de espaço de classificação. A cada espaço de classificação, uma rede neural artificial plana foi treinada para predizer a classe filha para um exemplo que já havia sido predito como pertencer a classe pai. O objetivo disso era fazer uma predição

adicional que fosse capaz de fazer anotações para as classes mais específicas a partir das anotações conhecidas para suas classes ancestrais.

Os autores utilizaram 44 espaços de classificação com um total de 131 classes. Para a avaliação da precisão do modelo de predição foi usado o índice de ATI (Averaged Tanimoto Index) de cada classe. Assim, as classes que tiveram um índice menor que um determinado limite foram eliminadas e as restantes foram utilizadas para fazer as predições adicionais para os espaços de classificação. Ao final foram obtidos 14 espaços de classificação contendo 45 classes.

Observa-se que, como no trabalho desenvolvido por Laegreid et al (2003), a estrutura de hierarquia das classes foi usada apenas na fase de pré-processamento para a criação do conjunto de dados de treinamento. Após essa etapa, para cada espaço de classificação é usado um algoritmo de classificação plana (FREITAS & CARVALHO, 2007).

Hayete e Bienkowska (2005) construíram uma árvore de decisão para cada termo do GO do mesmo domínio funcional. Para a avaliação do classificador, os autores utilizaram as medidas de sensibilidade e precisão. Nota-se, que o modelo criado tem como desvantagem a criação de várias árvores de decisão, o que pode tornar os resultados menos precisos comparados com o algoritmo que gera uma única árvore para o modelo.

4.2.2 Trabalhos Utilizando a Abordagem Classificação Hierárquica Local

Nesta seção serão apresentados alguns trabalhos envolvendo a estrutura hierárquica do tipo DAG utilizando a abordagem de classificação local. A Seção 3.2.2.1 descreve-se sobre os trabalhos envolvendo a abordagem de classificação local para cada nó. Já a Seção está dividida em 3.2.2.1 relata sobre os trabalhos categorizados na abordagem de classificação local em nós pais. Vale ressaltar que, até o conhecimento que se tem, nenhum trabalho foi desenvolvido utilizando a abordagem de classificação local por nível da hierarquia.

4.2.2.1 Classificação Local para cada Nó

Barutcuoglu et al (2006) utilizou o esquema hierárquico do GO para o desenvolvimento do trabalho. O conjunto de dados era formado por 105 nós e 1059 genes desconhecidos. Neste trabalho os autores treinaram um classificador (binário) individualmente para cada classe da hierarquia usando o algoritmo SVM. Nesta fase, os

autores não levaram em consideração a hierarquia das classes. Na sequência, para auxiliar na correção de erros entre todos os nós, os autores desenvolveram um *framework* hierárquico Bayesiano. Este *framework* tem com o objetivo combinar os classificadores baseados nos limites da hierarquia funcional de modo a corrigir predições incompatíveis na hierarquia de classe e melhorar o desempenho. Além disso, foi implementado de forma que as predições não fossem obrigatoriamente em nó folha.

A avaliação das predições foi medida através do cálculo da AUC. Os autores comparam os resultados obtidos pelo classificador SVM independentemente e quando combinado com o corretor Bayesiano. Nessa comparação concluíram que o *framework* Bayesiano ofereceu melhor resultado, mesmo para dados ruidosos em que o SVM pode causar predições incorretas.

Em Guan et al (2008) foi desenvolvido uma técnica que consiste de três classificadores, a saber: um único SVM, um SVM com correção hierárquica (para corrigir as predições baseadas no relacionamento hierárquico entre os termos do GO), e um classificador SVM combinado com uma rede Bayesiana.

Para fazer a correção hierárquica os autores usaram dois métodos: Markov Blanket Graphs (HIER-MB), e Breadth-First Search Graphs (HIER-BFS). Para avaliar os resultados os autores utilizaram a curva AUC. Em geral, para conjuntos de dados com menor quantidade de termos da GO o classificador único SVM e o SVM com correção hierárquica tiveram resultado superiores. Já para quantidades maiores de termos da GO a integração com a rede bayesiana apresentou melhores resultados que os outros dois métodos.

Em Jin et al (2008), os autores estudaram três algoritmos de classificação multiclasse baseado na abordagem de classificação, sendo dois deles locais utilizando os algoritmos SVM e Naïve Bayes, ambos utilizando a técnica *top-down*, e o outro classificador segue um mecanismo estocástico denominado de Random GO Walk (RGOW). A motivação para o desenvolvimento do RGOW é suavizar o problema de máximo local. O método desenvolvido RGOW transforma a estrutura hierárquica na forma de grafo direcionado em um grafo não direcionado. Assim, de acordo os autores, o algoritmo faz uma busca neste grafo e segue o caminho mais provável.

RGOW executa uma busca estocástica para encontrar as melhores classes baseada no algoritmo Metropolis Hastings com o procedimento recozimento simulado (Simulated Annealing).

Os autores desenvolveram o RGOW para explorar se procedimentos estocásticos podem ser usados para melhorar o problema de máximo local na busca *top-down* feita pelos

classificadores SVM e Naïve Bayes. Além disso, o sistema também produz uma distribuição probabilística nos rótulos folha.

Como medidas de avaliação, os autores utilizaram a medida baseada na distância semântica entre um par de termos da estrutura hierárquica. Além disso, utilizaram-se as medidas de precisão, revocação e a medida-F. Os autores fizeram a avaliação dos resultados baseado no gráfico, considerando o caminho de predição. Assim, caso uma classe fosse predita incorretamente, do ponto vista das medidas de avaliação anteriores, mais a predição estivesse acontecido no caminho válido essa seria considerada correta ou parcialmente correta.

Os autores avaliaram o método desenvolvido com a abordagem de classificação plana convencional, a qual apresentou resultados inferiores ao método desenvolvido.

4.2.3 Trabalhos Utilizando a Abordagem Classificação Hierárquica Global

Vens et al. (2008) desenvolveram um modelo de classificação hierárquica para a estrutura do tipo DAG usando a abordagem de classificação global.

Neste trabalho os autores abordam três técnicas de classificação: a classificação de uma única classe SC (*single-label classification*), a classificação hierárquica de uma única classe HSC (*hierarchical single-label classification*) e a classificação hierárquica de múltiplas classes HMC (*hierarchical multi-label classification*). Para o desenvolvimento destas técnicas os autores usam a indução de árvores de decisão e mostram como esse modelo poderá ser modificado para aplicação em estruturas hierárquica do tipo DAG.

Estas técnicas estão implementadas no sistema CLUS e consistem em gerar uma única árvore de decisão para toda a hierarquia. Esse algoritmo de indução da árvore de decisão é baseado no *framework* Predictive Clustering Trees (PCT).

Neste *framework*, uma árvore é vista como uma hierarquia de grupos, onde o nó raiz é um grupo que contém todos os dados e que de maneira recursiva e *top-down* é particionado em grupos menores. Este *framework* foi construído usando um algoritmo TDIDT (*Top-down induction of decision tree*s). A ideia geral consiste em particionar recursivamente o conjunto de dados de forma a reduzir a variância *intra-cluster*, maximizar a homogeneidade do agrupamento e melhorar o desempenho da predição. Mais detalhes sobre esse *framework* pode ser encontrado em Blockeel et al (2006).

Para a aplicação do PCT para a tarefa de classificação hierárquica multirrótulo, as classes das amostras são representadas através de vetores com valores booleanos (o *i-ésimo* valor da classe é 1 se a amostra pertencer a classe c_i caso contrário será 0).

Vens et al (2008) fizeram vários experimentos com as técnicas desenvolvidas. Para isso, 12 conjuntos de dados do organismo *Saccharomyces cerevisiae* foram usados. Foram feitas duas versões destes conjuntos, um usando o esquema de classificação MIPS's FunCat (estrutura de árvore) e a outra usando o esquema do GO (estrutura do DAG). Desta maneira, totalizou 24 conjuntos de dados. Vale ressaltar que os dados de entrada são os mesmos para ambas as versões, o que difere são os rótulos das classes.

Os resultados foram avaliados através das medidas de precisão e sensibilidade e pela curva AUC. Conforme os autores, o Clus-HMC teve um desempenho preditivo melhor comparado com as outras técnicas tanto para as estruturas de árvore como para DAG. Além disso, o tamanho da hierarquia quando usado o HCM é muito menor.

Em Aleksovski et al (2009), os autores estenderam o Clus HMLC desenvolvido por Vens et al (2008), utilizando outras medidas de distância. As medidas utilizadas pelos autores foram distância de Jaccard, SimGIC e ImageCLEF. Tais medidas foram implementadas no CLUS e aplicadas em dezesseis conjuntos de dados de genoma funcional, sendo doze desses seguindo o esquema de anotação FunCat (estruturados em árvore) e os quatro restantes no esquema GO (estruturados em DAG). Para avaliar o desempenho preditivo, os autores usaram as medidas de precisão e revocação juntamente com a curva precisão e revocação. Já para avaliar a significância desses resultados foi usado o teste estatístico de Friedman, o qual, conforme os autores não revelaram nenhuma diferença significativa nas duas medidas utilizadas. No que se refere os resultados obtidos pelas diferentes medidas, em relação à área abaixo da média da curva de precisão e revocação, a medida SimGIC teve o melhor resultado, seguida da distância euclidiana e na sequência da distância ImageCLEF. A distância de Jaccard teve o pior resultado. Já na avaliação da área média abaixo da curva de precisão e revocação a distância euclidiana teve o melhor resultado, seguida da medida SimGIC.

Alves et al. (2008) e Alves (2010) construíram um modelo de classificação hierárquica denominado de *Multi-Label Hierarchical Classification with an Artificial Immune System* (MHCAIS) o qual utiliza conceitos de um Sistema Imunológico Artificial (SAI). Este classificador hierárquico tem como objetivo descobrir conhecimento compreensível representado na forma de regras do tipo se-então.

O autor apresenta duas versões do MHCAIS: global e local. Na versão local é construído um classificador para cada classe do domínio. Neste classificador, o consequente das regras descobertas apenas diferencia se um exemplo (instância) pode ou não ser associado à classe para a qual aquele classificador foi treinado, ou seja, para cada nó da estrutura hierárquica. Já na versão global um único classificador é gerado para distinguir todas as classes do domínio da aplicação tratada. Nesta versão, uma ou mais classes são representadas no consequente.

O algoritmo MHCAIS possui dois procedimentos fundamentais, um para Extração Sequencial de Regras (*Sequential Covering*) e outro para Evolução das Regras (*Rule Evolution*).

O primeiro procedimento, Extração Sequencial de Regras, é usado em algoritmos de indução de regras para tratar problemas de predição de uma única classe. O autor, em seu trabalho, fez modificações no procedimento tradicional para poder ser utilizado na classificação hierárquica multiclasse.

O segundo procedimento, Evolução das Regras, é específico para algoritmos evolutivos ou relacionados, incluindo os SIAs baseados no princípio de seleção clonal. Este procedimento inicia com um conjunto vazio de regras, onde o mecanismo de Evolução de regras é iterativamente executado. Os anticorpos participam de um processo iterativo (evolução), cujo objetivo é encontrar as melhores regras (anticorpos) que formarão a solução (classificador) para o problema. Durante este processo, cada anticorpo é continuamente avaliado de acordo com sua capacidade de reconhecimento (classificação) dos antígenos da base de treinamento. Os melhores anticorpos são selecionados e adicionados ao conjunto de regras descobertas. Em seguida, os exemplos corretamente classificados pelas regras descobertas na iteração corrente são removidos. A versão global remove exemplos da base de dados de forma diferente da versão local. A construção do classificador termina quando o MHCAIS descobre o número de regras necessárias para classificar os exemplos da base de dados (ALVES, 2010).

No trabalho desenvolvido, o autor compara-se as abordagens propostas (MHCAIS global e local) com dois algoritmos: PART e Clus, ambos baseados em árvores de decisão que podem ser transformadas em regras se-então.

Para os experimentos computacionais foram utilizadas 10 bases de dados biológicos, sendo uma dela construída especificamente para os experimentos apresentados no trabalho. Já as bases restantes são de domínio público oriundas da GO. Os conjuntos de dados passaram por uma etapa de pré-processamento onde foram removidos termos da GO pouco frequentes

na base de dados. Este procedimento tem a finalidade de garantir um mínino de suporte estatístico no treinamento do algoritmo.

Os resultados são apresentados e analisados sob os critérios de correção preditiva com base na medida-F ou área da curva precisão e revocação e a simplicidade do conjunto de regras descobertas.

A correção preditiva (taxa de acerto) foi calculada considerando-se a base de teste (contendo exemplos não vistos durante a fase de treinamento).

Foram avaliadas duas funções de *fitness*: sensibilidade/especificicade e medida-F. Nos experimentos realizados o uso da função de *fitness* medida-F obteve melhor desempenho preditivo para a versão local e global do MHCAIS.

A avaliação dos resultados dos algoritmos MHCAIS Global/Local com o PART foi feita considerando o desvio-padrão, testes estatísticos de hipóteses de Wilcoxon. Já entre os algoritmos MHCAIS global, local e o Clus os critérios considerados para comparação foram a área da curva de AUPRC, número de regras descobertas e número total de condições presentes no conjunto de regras, além das análises estatísticas feitas através dos testes de Friedman e Wilcoxon.

Na comparação entre os resultados obtidos pelo algoritmo nos experimentos, o MHCAIS global e local foram superior ao algoritmo PART nos critérios de desempenho preditivo e simplicidade do conhecimento descoberto. Segundo os critérios estatísticos (desvio-padrão e teste de Wilcoxon) utilizados para análise, a diferença de desempenho foi significativa, principalmente no critério de simplicidade do conhecimento descoberto.

No comparativo com o algoritmo Clus, as duas versões do MHCAIS foram superadas no critério de desempenho preditivo (área da curva AUPRC). Porém, segundo o teste de hipóteses de Friedman, a diferença entre o Clus e o MHCAIS-global não foi significativa.

Já o resultado obtido pelo teste hipóteses de Wilcoxon, a diferença entre MHCAIS e Clus foi estatisticamente significativa. Considerando o critério do número de regras descobertas, o MHCAIS global não foi significativamente superior ao Clus segundo os testes de hipóteses de Friedman, porém o de acordo com teste de Wilcoxon existe diferença significativa de desempenho.

Em termos de número de condições, o MHCAIS global apresentou resultados significativamente superiores ao Clus-HMC, de acordo com o teste de Friedman e Wilcoxon. Porém, apresentou uma perda de desempenho preditivo em relação a simplicidade do conhecimento extraído dos dados.

Nos experimentos comparativos entre o MHCAIS e Clus-HMC Alves (2010) observou que possivelmente o modelo desenvolvido é sensível aos atributos previsores contínuos. Esta hipótese foi cogitada pelo autor uma vez que a única base de dados (dentre 10 bases usadas) que o MHCAIS foi ligeiramente superior ao Clus, visto que essa é formada por atributos categóricos (discretos). Deste modo, acredita-se que o método de busca local usado no trabalho não foi eficiente, embora tenha obtidos melhores resultados do que a versão do MHCAIS sem busca local. Além disso, de acordo com o autor, observou-se que o Clus-HMC tem um desempenho melhor com atributos contínuos pela sua capacidade de permitir que um mesmo atributo previsor ocorra múltiplas vezes na árvore construída. O MHCAIS não permite tal flexibilidade, pois cada atributo previsor ocorre uma única vez na regra descoberta pelo SIA.

Schietgat et al (2010) estenderam o trabalho de Vens et al(2008) construindo um classificador hierárquico multirrótulo denominado de Clus-HMC-ENS. O algoritmo desenvolvido por Vens et al (2008) gera uma única árvore que prevê, para um determinado gene, suas funções biológicas a partir de uma função de classificação. Já o Clus-HMC-ENS gera um conjunto de árvores. Através dos experimentos pode-se obter um resultado melhor comparado com o classificador Clus-HMC.

Otero et al. (2009) propôs um classificador hierárquico baseado no algoritmo de Colônia de Formigas denominado de hant-Miner (hierarchical classification Ant-Miner). A desvantagem do modelo é que ele não trabalha com classificação hierárquica multirrótulo.

Otero et al (2010) propôs uma extensão do classificador hant-Miner para tratar de problemas multiclasses denominado hmAnt-Miner (Hierarchical Multi-Label Classification Ant-Miner). Esse algoritmo descobre um único modelo de classificação global, através de uma lista de regras se-então, que pode prever todas as classes a partir da hierarquia de uma só vez. A fim de ter em conta as informações da hierarquia de classes o hmAnt-Miner emprega uma medida de distância baseada no procedimento de discretização dinâmica dos atributos contínuos e uma informação heurística na construção do grafo. Assim, a medida de entropia usada no hant-Miner é substituída pela medida de distância no hmAnt-Miner, que é uma medida mais adequada para a classificação hierárquica multirrótulo.

Bi & Kwok (2011) desenvolveram um classificador hierárquico que utiliza uma abordagem de estimativa de dependência de kernel (KDE) para reduzir a quantidade de classes e transformar o problema multirrótulo em um único rótulo. A estrutura hierárquica é preservada utilizando um algoritmo chamado de CSSA (Condensing Sort and Select Algorithm) o qual visa encontrar uma subárvore em uma árvore. Para tratar de problemas

estruturados em DAG os autores criaram o algoritmo CSSAG (CSSA for Graphs), que gera um subgrafo de um grafo. Assim, nos casos em que um nó tem mais de um nó ancestral é gerado um subgrafo replicando o nó filho.

4.3 TRABALHOS QUE USAM COMO TÉCNICA REDES NEURAIS E ESTRATÉGIA EVOLUCIONÁRIA

Jensen et al (2002) utilizaram várias Rede Neurais com uma camada escondida. O método foi aplicado no esquema de Riley e no esquema de Enzyme Commission (EC), cujas classes da hierarquia estão distribuídas na forma de árvore. O método desenvolvido utilizava apenas o primeiro nível da hierarquia, ou seja, o classificador desenvolvido é considerado plano, pois a hierarquia de classes não foi considerada. Para cada categoria funcional foram treinadas redes neurais. A avaliação do classificador foi feita utilizando a relação entre medida de sensibilidade e a taxa de falsos positivos.

Weinert & Lopes (2004) desenvolveram um classificador baseado em uma rede neural artificial MLP. O classificador foi aplicado em estrutura em árvore definida pela EC, porém, apenas no primeiro nível da hierarquia. Nesse trabalho, as classes foram consideradas como um problema de classificação plana, ou seja, a hierarquia de classes não foi considerada. Para avaliar o modelo os autores utilizaram como medidas a taxa de acerto e precisão e revocação.

Xiao et al. (2007) os autores construíram um classificador hierárquico local, usando uma rede neural (MLP) para classificar expressão emocional utilizando uma hierarquia com três níveis de e com 6 classes folhas (que são os tipos de emoção): raiva, tédio, medo, tristeza alegria e neutro.

Cerri & Carvalho (2011) criaram um classificador local denominado de HMC-LMLP (Hierarchical Multilabel Classification with Local Multi-Layer Perceptron) que treina uma rede neural perceptron multi-camadas (MLP), utilizando o algoritmo back-propagation, cada nível da hierarquia. Essa rede tem uma camada de entrada, uma camada escondida e uma camada de saída (classes). Após o treinamento de primeira rede, uma nova rede é criada para o segundo nível hierárquico. Os nós da camada de saída da rede anterior tornam-se a camada de entrada para esta nova rede, que será responsável para as previsões do segundo nível. Este procedimento é repetido para todos os níveis hierárquicos. Quando a rede está sendo treinada para um nível hierárquico, os pesos sinápticos das camadas utilizadas nos níveis anteriores não são ajustados, porque a sua adaptação já ocorreu anteriormente. Na fase de teste, para classificar uma nova instância, um limiar é associado a cada camada de saída da rede na

hierarquia. Cada saída (classe) cujo valor é maior ou igual ao seu limiar recebe o valor 1 e 0 caso contrário.

Vale ressaltar que, após a classificação é realizada uma correção de inconsistências que possam ter ocorrido durante o treinamento. Uma inconsistência ocorre quando uma classe em um determinado nível é predita, porém o mesmo não ocorre com a sua classe ancestral. Nessa fase, então, são removidas essas classes preditas, garantindo que apenas predições consistentes sejam feitas. De acordo com os autores, o HMC-LMLP pode sem aplicado em problemas multirrótulo. Foram feitos experimentos em 10 bases de dados estruturas em árvore, conforme o esquema do Funcat. Os resultados foram comparados com os algoritmos Clus-HMC, Clus-HSC e Clus-SC. Já a avaliação foi feita utilizando curvas de precisão e revocação e para verificar a significância dos resultados teste de Friedman e Nemenyi foram aplicados. Os resultados mostram que o HMC-LMLP apresentou resultados superiores aos classificadores Clus-HSC e Clus-SC em 80% dos casos.

No que se refere a algoritmos de classificação hierárquica usando a estratégia evolucionária não se encontrou, até o desenvolvimento do trabalho, nenhum algoritmo utilizando esta técnica, para tratar de problemas estruturados na forma de um DAG.

4.4 CONSIDERAÇÕES SOBRE TRABALHOS UTILIZANDO CLASSIFICAÇÃO GLOBAL APLICADO EM ESTRUTURAS DAG

Todos os trabalhos comentados, distribuídos nas abordagens de classificação, possuem suas particularidades, vantagens e desvantagens.

Os trabalhos desenvolvidos por Jensen et al. (2003), Laegreid et al. (2003) e Tu et al. (2004) os autores desconsideraram a hierarquia das classes, ou seja, o conceito de ancestral e descendente não foi aplicado. Dessa forma, pode-se dizer que o problema de classificação hierárquica foi transformado em problema de classificação plana, na qual foram utilizados algoritmos da classificação convencional.

Já os trabalhos de Barutcuoglu et al (2006), Guan et al (2008) e Jin et al (2008), os autores criaram um classificador local binário para cada nó. Nos dois primeiros trabalhos, os autores utilizaram uma rede bayesiana para garantir a hierarquia de classes. Porém, no contexto de classificação hierárquica local para cada nó, não há nenhuma garantia de que seja mantida a hierarquia de classes o que pode apresentar resultados não condizentes. Além disso, se ocorrer um erro em uma classe em um determinado nível, esse pode se propagar para as classes que estão nos níveis abaixo.

As técnicas de classificação hierárquica Clus-HMC e Clus-HSC desenvolvida por Vens et al (2008) e o MHCAIS desenvolvida por Alves (2010) apresentam algumas vantagens sobre as utilizadas por Jensen et al. (2003), Laegreid et al. (2003), Tu et al. (2004) e Barutcuoglu, Z. et al. (2006), Jin et al 2008 e Guan et al 2008. Primeiro, por se tratar de uma técnica que implementa a abordagem de classificação hierárquica global onde a hierarquia de classes é considerada como um todo. Dessa maneira, o modelo de predição construído considera, simultaneamente, todas as classes da hierarquia.

O algoritmo Clus (VENS et al, 2008), até onde se sabe, foi o primeiro algoritmo de classificação global, para estruturas do tipo DAG, desenvolvido especificamente para tratar problemas na área de bioinformática. Por ter sido desenvolvido baseado em uma árvore de decisão tem como vantagem a produção de modelos que podem ser interpretados por humanos.

Porém, o método de avaliação desenvolvido pelos autores, curvas AUCPR, utiliza de N limiares para a sua construção. Esses limiares são definidos em um intervalo de 0 e 1 e são comparados com a saída do classificador. Se a saída do classificador for maior ou igual ao valor do limiar considera-se que a classe foi predita. Quanto maior o limiar, maior será o valor obtido pela medida de precisão e menor o valor obtido pela medida de revocação (menos classes preditas) e assim, quanto menor o limiar, menor a precisão e maior a revocação (mais classes preditas).

Realizando este procedimento para cada limiar tem-se N predições e a obtenção do calculo da medida de precisão e revocação que ao serem interpolados geram a curva. Sendo assim, generalizando, pode-se dizer que se para obter uma resposta do Clus é necessário escolher um limiar.

Já o MHCAIS (ALVES, 2010) foi construído utilizando conceitos de um SIA e extração de regras do tipo se-então.

Comparando as duas técnicas de classificação global observa-se que o Clus-HMC apresenta melhores resultados, pois através dos experimentos feitos pelos autores, esse foi capaz de trabalhar melhor com atributos contínuos do que o algoritmo MHCAIS.

O classificador hierárquico desenvolvido por Otero et al (2010) também utiliza regras se-então utilizando a técnica de colônia de formigas e emprega uma medida de distância para avaliar o modelo.

Bi & Kowk transformam o problema de classificação hierárquica multirrótulo em um único rótulo. A hierarquia de classes, de acordo com os autores, é preservada, a classificação deixa ser multirrótulo o que pode melhorar os resultados dos experimentos. Para avaliar o

resultado do modelo os autores também utilizam as curvas AUCPR e comparam com o Clus-HMC.

Embora, particularmente, discorde-se do método de avaliação utilizado pelo algoritmo Clus, nesta tese será utilizado o mesmo para fazer as comparações com o algoritmo desenvolvido. Essa escolha deve-se por ser, atualmente, o classificador hierárquico multirrótulo mais utilizado no estado da arte na comparação de classificadores.

5 CLASSIFICADOR HIERÁRQUICO MULTIRRÓTULO USANDO UMA REDE NEURAL COMPETITIVA (MHC-CNN)

Neste Capítulo apresenta-se o algoritmo MHC-CNN que tem como objetivo auxiliar na tarefa de classificação de dados hierárquicos estruturados, principalmente, na forma de um DAG. Para tal, a Seção 5.1 descreve o algoritmo proposto seguindo os conceitos de uma rede neural competitiva. Já a Seção 5.2 apresenta uma variação do algoritmo MHC-CNN, no que se refere ao treinamento da rede neural, denominada de Multilabel Hierarchical Classification using a Evolutionary Strategical (MHC-ES). Já a Seção 5.3 apresenta as medidas utilizadas para avaliar o classificador.

5.1 DESCRIÇÃO DO ALGORITMO MHC-CNN

Uma das características de uma rede neural artificial competitiva é a sua habilidade de realizar mapeamentos que preservam a topologia entre os espaços de entrada e de saída. O processo de aprendizagem é baseado no aprendizado competitivo, onde os neurônios da camada de saída competem entre si para serem ativados de forma que apenas um neurônio de saída será o "vencedor". Os ajustes dos pesos sinápticos são feitos pelo neurônio que foi ativado e seus vizinhos.

O algoritmo MHC-CNN (BORGES & NIEVOLA, 2010), (BORGES & NIEVOLA, 2011), (BORGES & NIEVOLA, 20012c) é baseado em uma Rede Neural Artificial Competitiva. A Figura 26 mostra o modelo da rede neural. Essa rede consiste de duas camadas de neurônios. A camada de entrada é conectada a um vetor de entrada do conjunto de dados. O termo "Neurônios de Entrada" definido nesta figura representa todas as instâncias de entrada.

Cada neurônio na grade é conectado com todos os neurônios da camada de entrada. São apresentadas as instâncias de entrada à rede, e a cada instância apresentada tem-se uma região que é atualizada na grade a qual varia de uma instância apresentada para outra. Esses neurônios (camada de saída) da rede são criados conforme o número de classes que a base de dados possui.

Os neurônios são estimulados pelos exemplos de entrada durante o processo competitivo. Dessa maneira, serão considerados "vencedores" aqueles que mais se

assemelham a instância de entrada selecionada. Essa semelhança é feita através da utilização de medidas de distância.

Assim como em uma rede competitiva tradicional, como por exemplo, a rede de Kohonen, os neurônios da camada de saída são dispostos em uma grade, a qual pode ser retangular, hexagonal, entre outras, e que tem como objetivo apresentar a topologia da rede. No algoritmo MHC-CNN esta topologia tem a estrutura de um grafo acíclico dirigido como mostra a Figura 27, onde cada neurônio está conectado com seus neurônios ancestrais (pais) e descendentes (filhos).

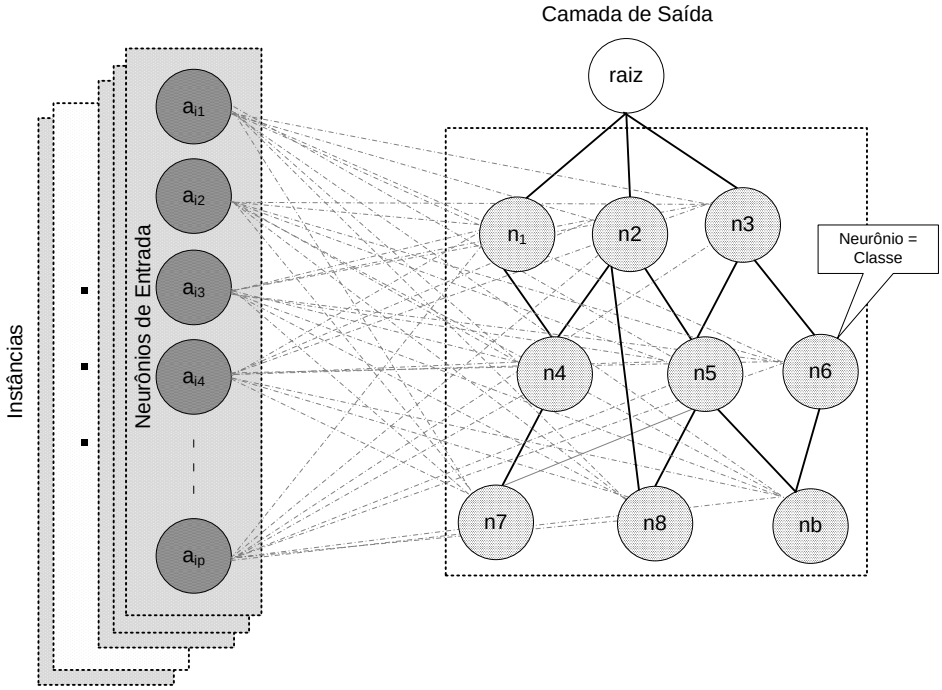


Figura 26 – Modelo do MHC-CNN.

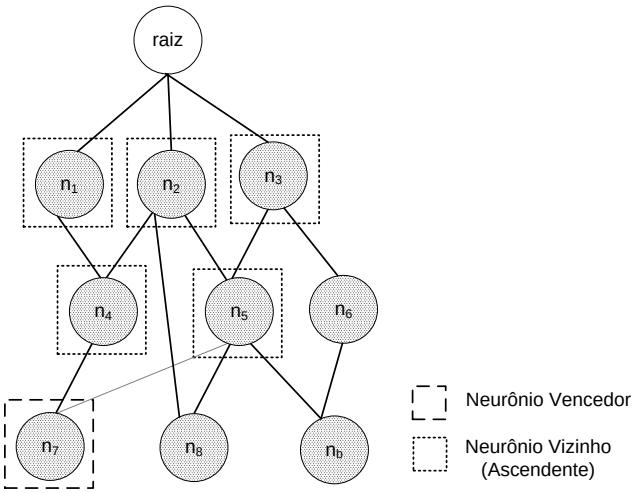


Figura 27 – Topologia de Saída do MHC-CNN.

Primeiramente, deve-se determinar a quantidade de épocas ep para o treinamento da rede neural e a taxa de aprendizagem (inicial μ_i e final μ_f) a qual irá decrescer exponencialmente ao longo desse treinamento.

O conjunto de pesos sinápticos é inicializado aleatoriamente. O processo de treinamento da rede é dividido em três fases, assim como em uma rede competitiva tradicional: Competição, Cooperação e Adaptação.

Competição

Uma instância e_i do conjunto de dados de entrada $BD_{trein} = [e_1\ e_2\ e_3 ... e_q]$ de dimensão m é selecionada. Cada elemento de BD_{trein} é formado por atributos, $v_i = [a_{1i}, a_{2i}, ..., a_{li}]$ em que a_{jk} é o j-ésimo atributo, $1 \leq j \leq l$, $l-1$ é o número de atributos de entrada e $i \in \mathrm{N} : 1 \leq i \leq q$, a_{li} representa o atributo classe. Em um problema multirrótulo o atributo a_{li} é composto por vários rótulos (classes).

A rede neural é criada conforme as informações obtidas nos dados de entrada. Tem-se, então, a rede neural formada por uma camada. O número de neurônios de entrada é igual ao número de atributos de entrada. Os neurônios da camada de saída são representados por $RN = [n_1\ n_2\ n_3 ... n_b]$ onde b é o número de classes que existe na hierarquia de classes. Cada neurônio da camada de saída é constituído de pesos sinápticos $n_i = [p_{1i}, p_{2i}, ..., p_{(l-1)i}]$ em que $i \in \mathrm{N} : 1 \leq i \leq (l-1)$, $(l-1)$ é o número de atributos de entrada das instâncias.

Para encontrar o vetor de entrada e_i que mais se aproxima do vetor de pesos sinápticos n_j, são utilizadas medidas de distâncias. A medida de distância escolhida foi a Euclidiana (Equação 43).

A distância Euclidiana é a raiz quadrada da soma dos quadrados das diferenças de valores para cada atributo.

$$d_{ik} = \sum_{i=1}^{l-1} \sqrt{(e_{ij} - n_{ijk})^2} \qquad \textbf{Equação 43}$$

em que e_i é uma instância de entrada, n_{ijk} é o k-ésimo neurônio da camada de saída, $(l-1)$ é o número de atributos previsores de entrada e k é quantidade de neurônios da camada de saída.

O próximo passo do algoritmo é identificar o(s) neurônio(s) de saída que apresenta(m) a(s) menor(es) distância(s) conforme a quantidade de classes alvo que a instância e_i possui

(Equação 44). Por exemplo, se e_i tem três classes objetivos deve-se identificar os três neurônios que apresentaram as menores distâncias, pois o objetivo é encontrar o conjunto de pesos sinápticos que mais se aproxima da instância de entrada selecionada. Dessa forma, tem-se o(s) neurônio(s) vencedor(s) do processo de competição da época considerada.

$$nv = \arg\min(d_i)$$ **Equação 44**

Considerando que no processo de competição tem-se uma região de vizinhança é necessário atualizar os pesos sinápticos do neurônio vencedor e de seus vizinhos. No algoritmo MHC-CNN, o critério de vizinhança é determinado a partir da relação existente entre o neurônio vencedor e seus ancestrais (pais do neurônio vencedor) e descendentes (filhos do neurônio vencedor). Estas informações são obtidas por meio do relacionamento existente na hierarquia de classes.

Assim, o neurônio da rede RN que apresentar a menor distância em relação a instância de entrada selecionada será considerado o vencedor e estimulará a sua vizinhança conforme a topologia da rede.

Cooperação

O neurônio vencedor é o neurônio filho na vizinhança topológica o qual tende a excitar os seus neurônios ancestrais que estão mais próximos do que aqueles que estão mais distantes. Sendo assim, nessa fase localiza-se a região da vizinhança topológica que terá os pesos dos neurônios atualizados.

Adaptação

A terceira fase tem como objetivo o processo adaptativo dos pesos sinápticos. Como o objetivo do treinamento é deixar os vetores de pesos mais próximos das instâncias de entrada, os pesos dos neurônios precisam ser ajustados para que, posteriormente, haja uma melhor classificação da instância. Sendo assim, se os rótulos de classe da instância de entrada e_i forem iguais ao do neurônio vencedor os pesos desse serão ajustados de maneira que fiquem mais próximos da instância e_i. Neste caso, tem-se uma predição correta. Já se os rótulos de classe da instância e_i forem diferentes do neurônio vencedor, os pesos desse serão atualizados para ficarem mais afastados desta instância, pois a classe predita está incorreta. Ou seja, durante o treinamento, o algoritmo ajusta os pesos do neurônio vencedor e de seus ancestrais, fazendo a comparação pelo identificador do rótulo de cada instância de entrada com a saída desejada.

A Equação 45 mostra o cálculo para a atualização dos pesos do neurônio verdadeiro e do falso vencedor.

$$n_{ijk}(t+1) = \begin{cases} (n_{ijk} + (e_i(t) - n_{ij}(t)) * Ap(t) * Dist(t)), \ se\ a\ classe\ for\ correta \\ (n_{ijk} - (e_i(t) - n_{ij}(t)) * Ap(t) * Dist(t)), \ caso\ contrário \end{cases} \quad \textbf{Equação 45}$$

em que n_{ijk} é o peso do atributo do neurônio da camada de saída na iteração (t) entre o neurônio de entrada (instância) $e_i(t)$ e o neurônio k.

$Ap(t)$ é a taxa de aprendizagem para o instante de tempo $(t+1)$ a qual é obtida pela fórmula apresentada na Equação 46.

$$Ap(t) = (\mu_i - \mu_f) * e^{-\frac{t_{atual}}{C}} \quad \textbf{Equação 46}$$

em que μ_i é a taxa de aprendizagem inicial, μ_f é a taxa de aprendizagem final, t_{atual} é a iteração atual e C é uma constante definida para que a função exponencial decresça lentamente.

$Dist(t)$ é a distância do neurônio vencedor até o seu ancestral que está sendo atualizado e é obtida pela fórmula mostrada na Equação 47.

$$Dist(t) = 1/(k+1) \quad \textbf{Equação 47}$$

sendo k a distância (ligações em nós) entre o(s) neurônio(s) vencedor(es) e o(s) neurônio(s) que terá(ão) seus pesos ajustados.

A atualização dos pesos dos neurônios ancestrais referentes aos neurônios pertencentes a classe correta e incorreta se dará da mesma maneira apresentada nas equações anteriores (Equação 45, Equação 46 e Equação 47).

Após as atualizações dos pesos dos neurônios, uma nova instância é selecionada e todo o procedimento descrito anteriormente se repete até que todas as instâncias sejam selecionadas. Ao final, tem-se concluído a primeira época da etapa de treinamento. Novamente, todo o procedimento descrito é repetido até a execução de todas as épocas. Tem-se então a fase de treinamento concluída.

Na última iteração desta fase obtêm-se o conjunto de pesos sinápticos dos neurônios que será utilizado na fase de teste do algoritmo. A Tabela 6 apresenta o algoritmo MHC-CNN de forma simplificada.

O teste do algoritmo é feito de maneira semelhante ao treinamento. Da mesma maneira que a base de dados de treinamento, a base de dados de teste é formada por instâncias BD_{test}, representadas por $BD_{test} = [x_1\ x_2\ x_3\ ...\ x_g]$ onde g é o total de instâncias de teste. Cada

instância x_i é formada por atributos, $x_i = [a_{1i}, a_{2i}, ..., a_{li}]$ em que $i \in \mathrm{N} : 1 \leq i \leq g$, a_{li} representa o atributo classe e l é o total de atributos.

Tabela 6 – Passo para o treinamento do algoritmo MHC-CNN.

ENTRADA DO ALGORITMO

- Conjunto de dados de treinamento $BD_{trein} = [e_1\ e_2\ e_3 ... e_q]$ de dimensão m.

PASSO 1: INICIALIZAÇÃO

- Inicializar a taxa de aprendizagem inicial μ_i e a final μ_f.
- Determinar o número de épocas ep.
- Inicializar os pesos sinápticos da rede: $RN = [n_1\ n_2\ n_3\ n_b]$ onde b é o número de classes que existe na hierarquia de classes.

PASSO 2: CRITÉRIO DE PARADA

- Número de épocas.

PASSO 3: TREINAMENTO

- Selecionar uma instância e_i do conjunto de dados de entrada $BD_{trein} = [e_1\ e_2\ e_3 ... e_q]$.

FASE COMPETIVA:

- Calcular a distância entre a instância v_i com os neurônios da rede $RN = [n_1\ n_2\ n_3\ n_b]$.
- Encontrar o(s) neurônio(s) n_j que apresentaram a(s) menor(es) distância(s), os quais serão considerados vencedores.

FASE COOPERAÇÃO:

- Encontrar os ancestrais do neurônio j.

FASE ADAPTAÇÃO:

- Atualizar os pesos sinápticos do neurônio(s) vencedor(es) e de seus ancestrais:
$$n_{ijk}(t+1) = \begin{cases} (n_{ijk} + (v_i(t) - n_{ij}(t)) * Ap(t) * Dist(t)), \ se\ a\ classe\ for\ correta \\ (n_{ijk} - (v_i(t) - n_{ij}(t)) * Ap(t) * Dist(t)), \ em\ caso\ contrário \end{cases}$$
- $Dist(t) = 1$ se a classe verdadeira for igual a classe predita.
- Calcular a taxa de aprendizagem $Ap(t) = (\mu_i - \mu_f) * e^{-\frac{t_{atual}}{C}}$ em que t_{atual} é a iteração atual e C é uma constante definida para que a função exponencial decresça lentamente.

PASSO 4: ATUALIZAÇÃO

- Atualizar a taxa de aprendizagem $Ap(t)$.
- Inicia o PASSO 1.

SAÌDA DO ALGORITMO

- Conjunto de pesos considerado adequado.

A Tabela 7 mostra o procedimento de teste do algoritmo. Observa-se que esse é parecido com o treinamento do algoritmo. A principal diferença é que nesta fase os pesos dos neurônios da camada de saída são fixos, provenientes da última época da fase de treinamento. O algoritmo apenas precisa utilizar este conjunto de pesos para testar com a base de dados de teste.

Tabela 7 – Passos para o teste do algoritmo MHC-CNN.

ENTRADA DE DADOS
- Conjunto de dados de teste BD_{test}, representadas por $BD_{test} = [x_1\ x_2\ x_3\ ...\ x_g]$ onde g é o total de instâncias de teste.
PASSO 1: CRITÉRIO DE PARADA
- Até que todas as instâncias tenham sido selecionadas e testadas.
PASSO 3: TESTE
- Selecionar uma instância x_i do conjunto de dados de entrada $BD_{test} = [x_1\ x_2\ x_3\ ...\ x_g]$. - Calcular a distância entre a instância x_i com os neurônios da rede $RN = [n_1\ n_2\ n_3\ n_b]$. - Encontrar o(s) neurônio(s) n_j que apresentaram a(s) menor(es) distância(s), os quais serão considerados vencedores. - Encontrar os ancestrais do neurônio j. - Comparar predição. - Atribuir resultado para a matriz de confusão.
SAÍDA DO ALGORITMO
- Taxa de acerto obtido pelo algoritmo.

O procedimento Comparar Predição, no PASSO 3, é o mais complexo desta fase, pois é a partir desse que se obtém a taxa de acerto do algoritmo. Detalhes sobre como foi feita a avaliação são descritos na Seção 5.2.

5.2 DESCRIÇÃO DO ALGORITMO MHC-ES

O algoritmo MHC-ES é uma versão modificada do algoritmo MHC-CNN o qual consiste em treinar a rede neural usando a estratégia evolucionária utilizando a técnica (μ + λ).

A ideia geral desse classificador é utilizar os passos da estratégia evolucionária para a atualização dos pesos da rede neural, ou seja a mudança de classificador se encontra na fase de treinamento. Já a fase de teste do algoritmo permanecerá igual ao do classificador MHC-CNN.

Ao invés de ser criada uma rede neural, aqui serão criadas *N* redes o qual chamar-se-á de indivíduos ou cromossomos conforme o nome usado na técnica original. A Figura 28 mostra um exemplo de um indivíduo criado a partir da hierarquia de classes. O modelo do algoritmo MHC-ES, formado por *N* indivíduos é mostrado na

Figura 29.

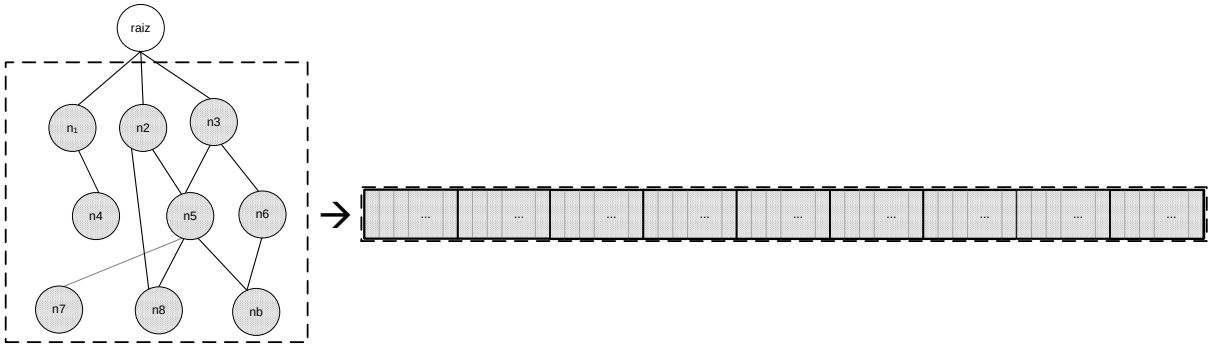


Figura 28 – Transformação da Hierarquia de Classes em Indivíduo da ES.

Assim como no MHC-CNN, devem-se determinar alguns parâmetros de entrada. No MHC-ES determina-se o tamanho da população P, a qual têm seus valores inicializados aleatoriamente, e a quantidade de gerações G.

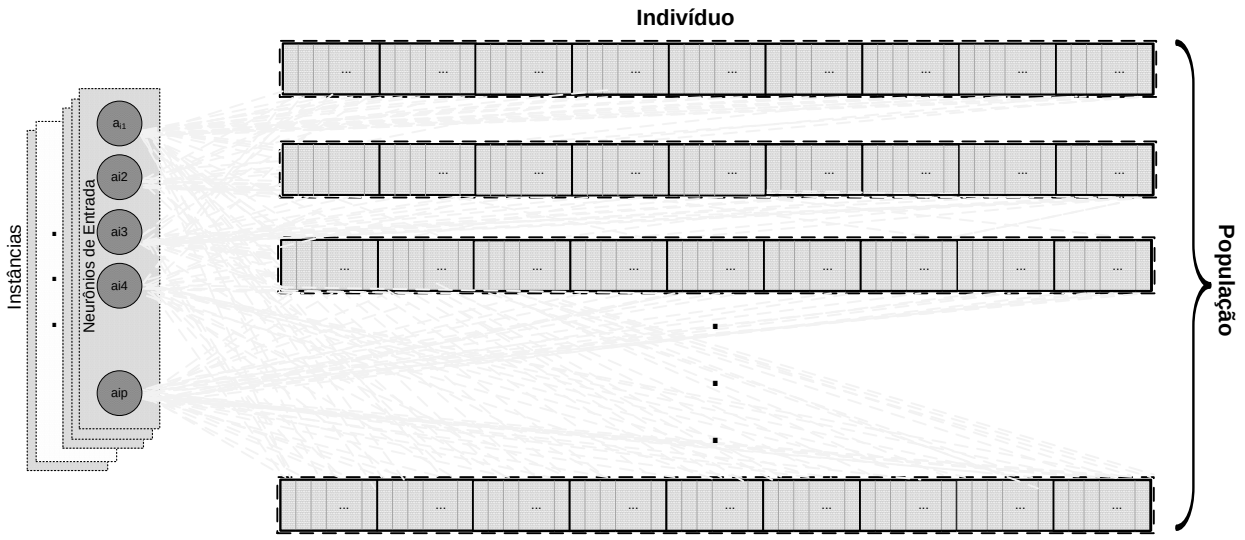


Figura 29 – Modelo do MHC-ES.

A população de indivíduos é criada conforme o parâmetro de entrada P bem como a quantidade de gerações G. Já o tamanho do indivíduo é conforme a quantidade de classes existentes na hierarquia de classes.

O próximo passo é calcular a distância euclidiana (Equação 43) entre cada instância de entrada com todos os indivíduos da população. Na sequência, assim como no algoritmo MHC-ES, obtêm-se a taxa de acerto de cada individuo, ou seja, o *fitness* de cada indivíduo.

Baseado no *fitness*, os indivíduos são selecionados através do método da roleta, para que na sequência possam sofrer a recombinação e a mutação.

A recombinação é calculada da seguinte maneira: dois indivíduos são selecionados $pai1$ e $pai2$. Esses dois indivíduos darão origem a dois indivíduos descendentes $filho1$ e $filho2$. O cálculo da recombinação é mostrado na Equação 48.

$$filho1 = pai1 * c + (pai2 * (1 - c)) \qquad \textbf{Equação 48}$$

$$filho2 = (pai1 * (1 - c)) + pai2 * c$$

em que c é uma constante.

Posteriormente, esses indivíduos sofrerão a mutação. O tipo de mutação utilizado foi mostrado na Equação 7 na Seção 2.2.2.

5.3 MEDIDAS DE AVALIAÇÃO

Três abordagens de medidas de avaliação foram utilizadas para reportar o desempenho preditivo das amostras: medida de distância dependente de profundidade, medida baseada nas relações de ancestralidade e descendência e a curva de precisão e revocação. A escolha por essas medidas está em avaliar o resultado da classificação de diferentes maneiras.

5.3.1 Medida Baseada em Distância Dependente de Profundidade

Quando se avalia o resultado de uma predição hierárquica três situações podem ocorrer: a predição pode ser correta, parcialmente correta ou incorreta.

Para um melhor entendimento de cada destas situações, de como foi feita a avaliação e quais foram o critérios adotados, até o presente momento, será exemplificado cada uma das situações.

i) Predição Correta

Considera-se como correta duas possibilidades de predição. A primeira possibilidade ocorre em casos onde o algoritmo acerta totalmente o caminho de predição e a classe predita é igual à classe verdadeira como mostrada na Figura 30.

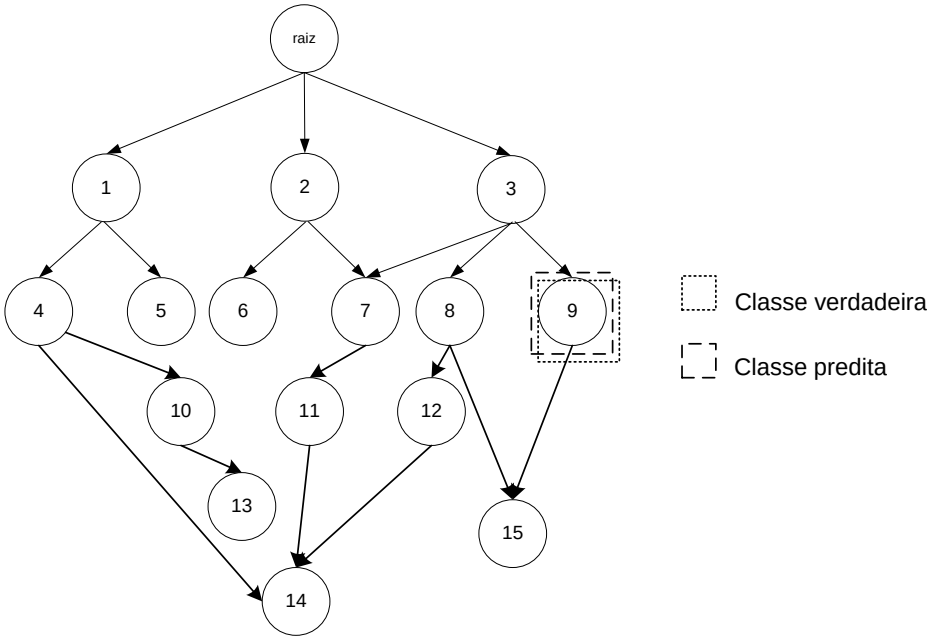


Figura 30 - Exemplo de Predição Correta – 1ª Possibilidade.

A segunda possibilidade de predição ocorre quando o algoritmo ao predizer uma determinada classe se torna muito específico. A Figura 31 mostra esta possibilidade em que a classe verdadeira é representada pelo nó "9" no DAG, porém, o algoritmo faz a predição como sendo a classe representada pelo nó "15". Neste caso também considera-se como predição correta.

ii) Predição Parcialmente Correta

Um exemplo de predição parcialmente correta é mostrado na Figura 32. Neste caso, a classe verdadeira é representada pelo nó "14" no DAG, porém, o algoritmo faz a predição como sendo a classe representada pelo nó "2". Observa-se que o nó predito é pai do nó verdadeiro. Embora o algoritmo esteja no caminho correto de predição essa ocorre antes de encontrar a classe verdadeira no grafo. Sendo assim, pode dizer que a predição foi

parcialmente correta, pois o algoritmo estava no caminho correto de predição apenas essa ocorreu antes.

Para a avaliação deste tipo de situação, atribui-se um acerto parcial baseado na distância em nós da classe predita até a classe verdadeira em relação a máxima distância possível.

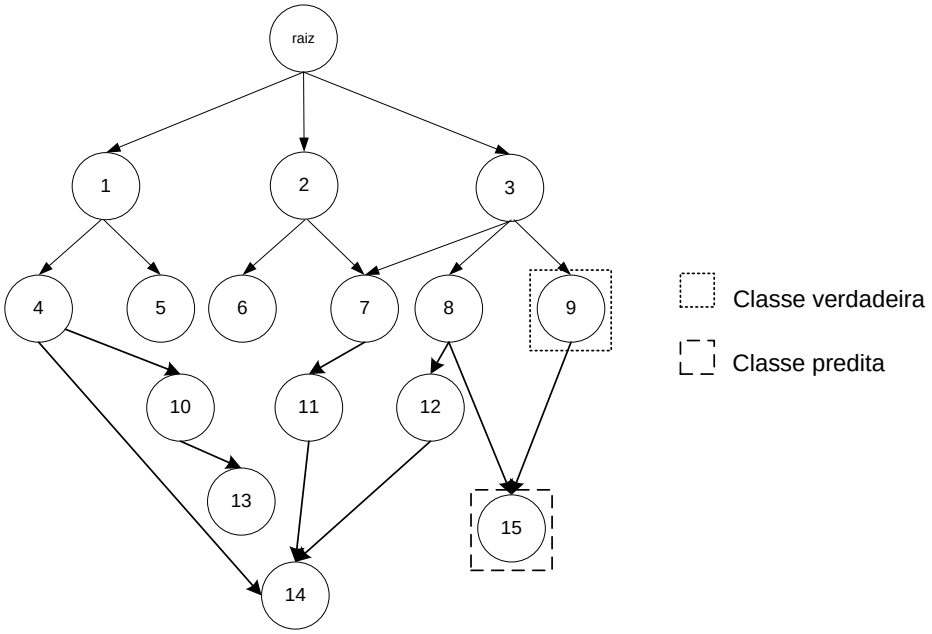


Figura 31 - Exemplo de Predição Correta – 2ª Possibilidade.

Uma instância predita cuja classe está em níveis mais altos tende a ser mais fácil a sua classificação do que uma classe que está em níveis mais profundos. Dessa maneira, para avaliar o algoritmo atribui-se pesos para a predição. Esses pesos são gerados de forma linear baseado no nível da classe, ou seja, classes em níveis mais próximos da raiz terão pesos maiores do que as classes em níveis mais profundos.

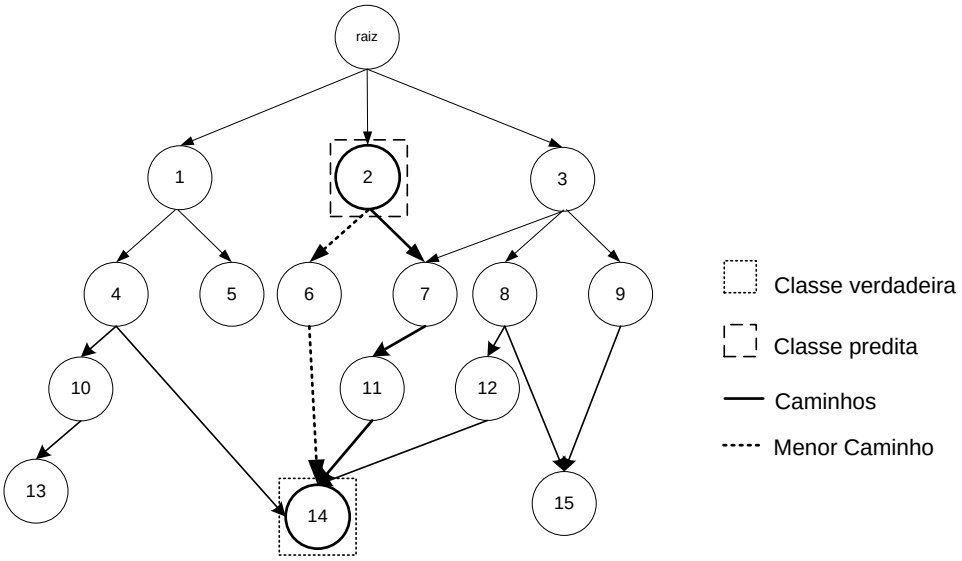


Figura 32 - Exemplo de Predição Parcialmente Correta.

Como uma hierarquia, em seu caso mais complexo, é um grafo acíclico dirigido em que um nó pode ter vários nós ancestrais, adotou-se o critério de menor caminho para avaliar o resultado da predição. Para a avaliação parcial dessa predição, primeiramente, verifica-se quais os caminhos possíveis entre a classe predita e a classe verdadeira, pois em um grafo pode existir vários caminhos de um nó a outro. Utilizando o critério de menor caminho define-se qual caminho será avaliado e atribuído pesos para as classes, baseado em seus níveis. Assim, o menor caminho no exemplo é entre as classes 2→6→14. Considerando que, neste caminho, a classe predita está no primeiro nível e a verdadeira no terceiro nível atribui-se pesos inversamente proporcionais aos níveis das classes, ou seja, a classe "2" passa a ter um peso 3 vezes maior que a classe "14". A Equação 49 mostra o calculo que é feito nessa etapa.

$$1p + 2p + ... + np = 1 \qquad \textbf{Equação 49}$$

em que p é o peso e n é a quantidade de níveis existentes no menor caminho. A taxa de acerto da predição é a soma dos pesos das classes preditas corretamente, ou seja, a classe predita e a as suas classes ancestrais. Aplicando na fórmula tem-se $p = 0{,}16$. Assim, o peso da classe "14" é 0,16, da classe "6" é 0,33 e da classe "2" 0,5. Logo, a taxa de acerto dessa amostra será 16%.

iii) Predição Incorreta

Uma predição é considerada incorreta nos casos onde o algoritmo erra o caminho de predição como mostrado na Figura 33. Observa-se que a classe verdadeira está representada pelo nó "13" no DAG, porém, o algoritmo prediz incorretamente como sendo o nó "7".

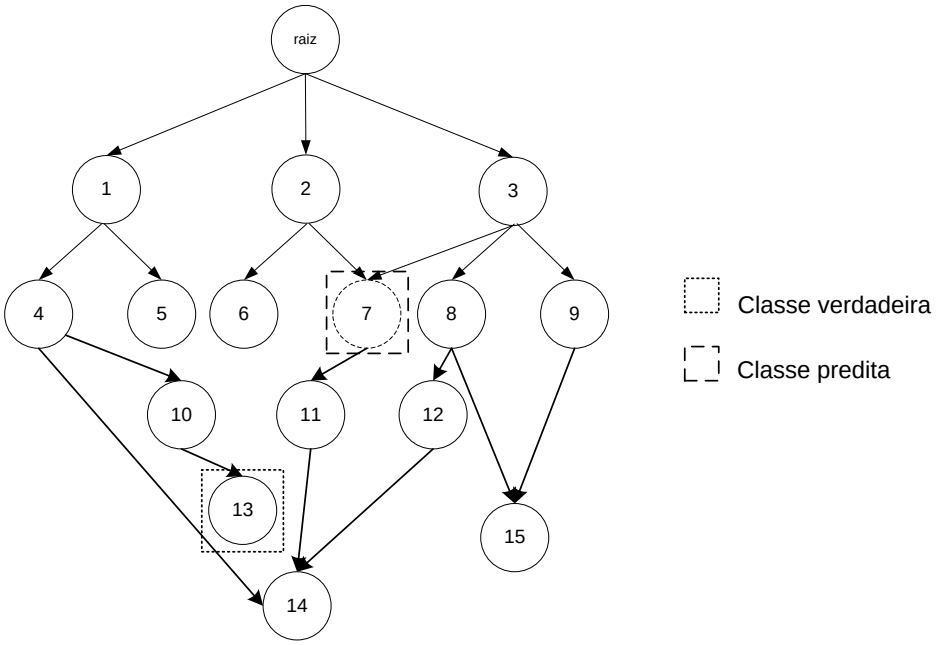


Figura 33 - Exemplo de Predição Incorreta.

5.3.2 Medida Baseada na Relação de Ancestralidade

A medida utilizada baseada nas relações de ancestralidade utilizada foi desenvolvida por Kiritchenko et.al. (2004) é citada na Seção 3.4.3 através da Equação 31 e Equação 32. Considerando o exemplo de hierarquia mostrado na Figura 34, obtêm os seguintes dados:

- Ancestrais da Classe Predita (incluindo a própria classe): 7, 2, 3.
- Ancestrais da Classe Verdadeira (incluindo a própria classe): 14, 4, 1, 6, 2, 11, 7, 12, 8, 3.
- Intersecção das Classes Predita e Verdadeira: 7,2, 3.

Assim, a medida de sensibilidade hierárquica será igual a $Sh = 0{,}3$ e a medida e precisão hierárquica será igual a $Ph = 1$.

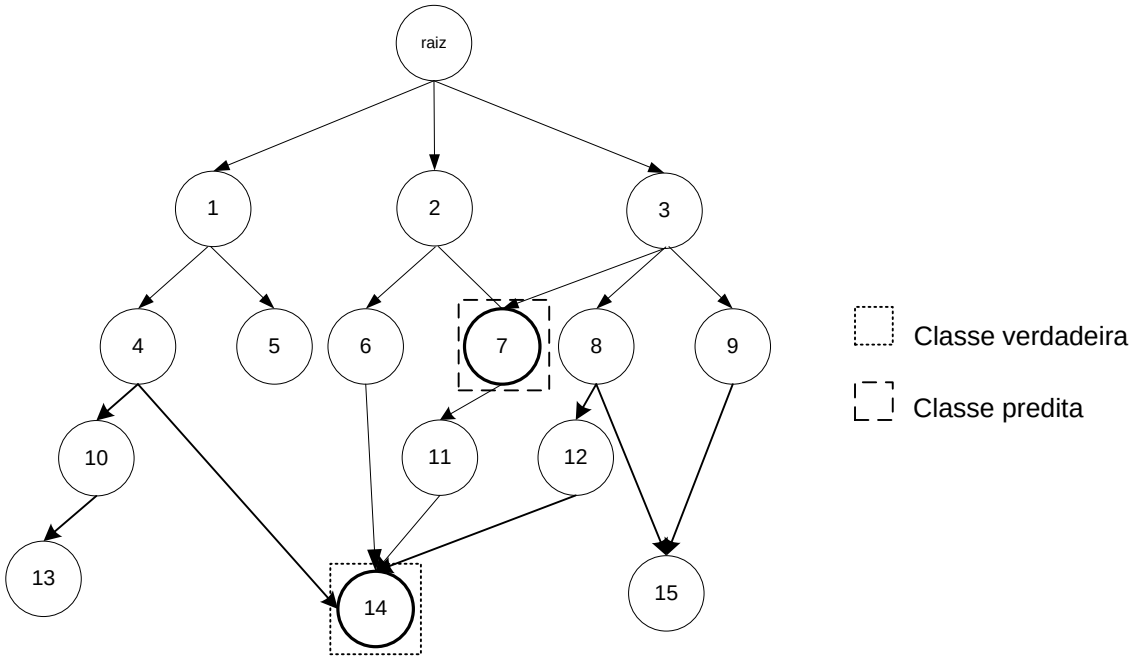


Figura 34 – Exemplo de Medida baseada na Hierarquia.

5.3.3 Medida Baseada na Curva de Precisão e Revocação

Esta medida funciona conforme citado na Seção 3.4.4, em que um conjunto de limiares entre 0 e 1 são selecionados. Cada limiar corresponde a um ponto no espaço da curva PR e variando esses limiares obtêm-se a curva PR.

Os resultados fornecidos pelos *scripts* do algoritmo Clus utilizam a seguinte sintaxe: T(*valor*), em que *valor* é um número inteiro como, por exemplo, T(20) que corresponde ao limiar 0.2.

Os limiares definidos como padrão do *script* iniciam em 0 e tem um incremento de 0,02 até o último limiar que é igual a 1, totalizando assim 51 limiares.

Para facilitar a comparação entre os resultados dos algoritmos será padronizando a representação do limiar conforme o exemplo apresentado.

6 EXPERIMENTOS

Neste Capítulo são apresentados os experimentos realizados e os resultados obtidos com os classificadores propostos, MHC-CNN e HMC-ES, investigando sua eficiência em relação a outros algoritmos da literatura. A Seção 6.1 faz uma breve descrição sobre os experimentos iniciais que foram feitos utilizando conjunto de dados estruturados na forma de árvore. A Seção 6.2 descreve as bases de dados utilizadas para a realização dos experimentos. A Seção 6.2 apresenta a metodologia usada para o desenvolvimento dos experimentos. Na Seção 6.3 são apresentados os resultados obtidos com os experimentos realizados. Por fim, a Seção 6.4 apresenta um comparativo geral dos resultados obtidos.

6.1 EXPERIMENTOS INICIAIS

Para a realização dos experimentos iniciais foram utilizadas 8 bases de dados do Funcat, as mesmas utilizadas por Holden & Freitas (2009). Essas são bases de dados estruturadas em árvore e possuem apenas um atributo meta. Devido a está característica, os classificadores foram primeiramente desenvolvidos para tratar de problema simples rótulo os quais foram denominados de Hierarchical Classification using a Competitive Neural Network (HC-CNN) (BORGES & NIEVOLA, 2012a), (BORGES & NIEVOLA, 2012b) e Hierarchical Classification using Evolutionary Strategical (HC-ES).

Como o objetivo do trabalho é utilizar bases de dados estruturados em DAG, a metodologia dos experimentos bem como os resultados e comparações com outros algoritmos são apresentados no Apêndice A.

6.2 BASES DE DADOS

Para a realização dos experimentos foram utilizadas dez bases de dados GO. Essas bases são as mesmas utilizadas por Vens et. al (2008), Alves et. al (2010), Otero et al. (2011). A Tabela 8 apresenta algumas características dessas bases as quais podem ser encontradas em http://www.cs.kuleuven.be/~dtai/clus/hmcdatasets.

Com o objetivo de melhor descrever a origem dos dados das bases de dados, a Tabela 9 faz uma breve descrição apresentando tais informações.

Tabela 8 – Características das Bases de Dados GO.

Base de Dados	Quant. Amostras	Quant. Atributos	Quant. Classes	Quant. Max. Níveis	Quant. Min/Max Classes por Amostras	Quant. Min/Max Amostras por Classe
Cellcycle (SPELLMAN et al., 1998)	3751	77	4125	13	3/28	0/785
Church (ROTH et al., 1998)	3749	27	4125	13	3/28	0/786
Derisi (DERISI et al, 1997)	3719	63	4119	13	3/28	0/781
Eisen (EISEN et al., 1998)	2418	79	3573	13	3/28	0/492
Expr (CLARE, 2003)	3773	551	4131	13	3/28	0/789
Gasch1 (GASCH et al., 2000)	3758	173	4125	13	3/28	0/786
Gasch2 (GASCH et al., 2001)	3773	52	4131	13	3/28	0/789
Pheno (CLARE, 2003)	1586	69	3127	13	3/21	0/388
Seq (CLARE, 2003)	3900	478	4133	13	3/28	0/791
Spo (CHU et al., 1998)	3697	80	4119	13	3/28	0/775

Tabela 9 – Informações sobre as Bases de Dados GO.

Base de Dados	Origem dos Dados	Tipos de Dados
Cellcycle	Dados biológicos de expressão gênica, obtidos pela técnica de microarranjo.	Atributos numéricos e alguns faltantes.
Church		Atributos numéricos, um categórico e alguns faltantes.
Derisi		Atributos numéricos.
Eisen		Atributos numéricos e alguns faltantes.
Gasch1		Atributos numéricos e alguns faltantes.
Gasch2		Atributos numéricos e alguns faltantes.
Spo		Atributos numéricos e alguns faltantes.
Expr	Concatenação das bases de dados de expressão gênica, obtidas pela técnica de microarranjo.	Atributos numéricos e alguns faltantes.
Pheno	Informações referentes ao fenótipo de leveduras mutantes. Certos genes são desativados ou removidos para que se possa avaliar o comportamento do organismo a este procedimento. Os dados foram extraídos das fontes EUROFAN (OLIVER, 1996), MIPS (MEWES et al., 1999) e TRIPLES (KUMAR et al., 2000).	Atributos categóricos.
Seq	Dados estatísticos da sequência de aminoácidos na construção dos atributos previsores. Dentre as características, pode-se citar: taxa de aminoácidos, tamanho da sequência, peso molecular. Alguns dos atributos foram obtidos através do programa ProtParam e outros do MIPS.	Atributos numéricos, discretos e alguns faltantes.

6.3 METODOLOGIA UTILIZADA PARA A REALIZAÇÃO DOS EXPERIMENTOS

Basicamente a metodologia utilizada para a realização dos experimentos é formada por três etapas principais: pré-processamento, classificação hierárquica e a avaliação dos

resultados obtidos conforme pode ser visualizado na Figura 35. Na sequência é descrita detalhadamente cada umas dessas etapas.

6.3.1 Pré-Processamento das Bases de Dados

Nesta etapa vários ajustes foram feitos para que fosse possível a utilização das bases de dados no classificador hierárquico proposto. Os passos realizados nesta etapa são mostrados na Figura 36.

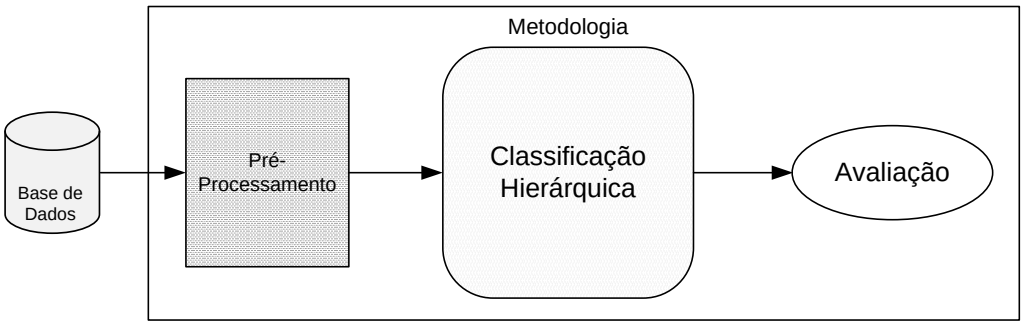


Figura 35 – Metodologia para Realização dos Experimentos.

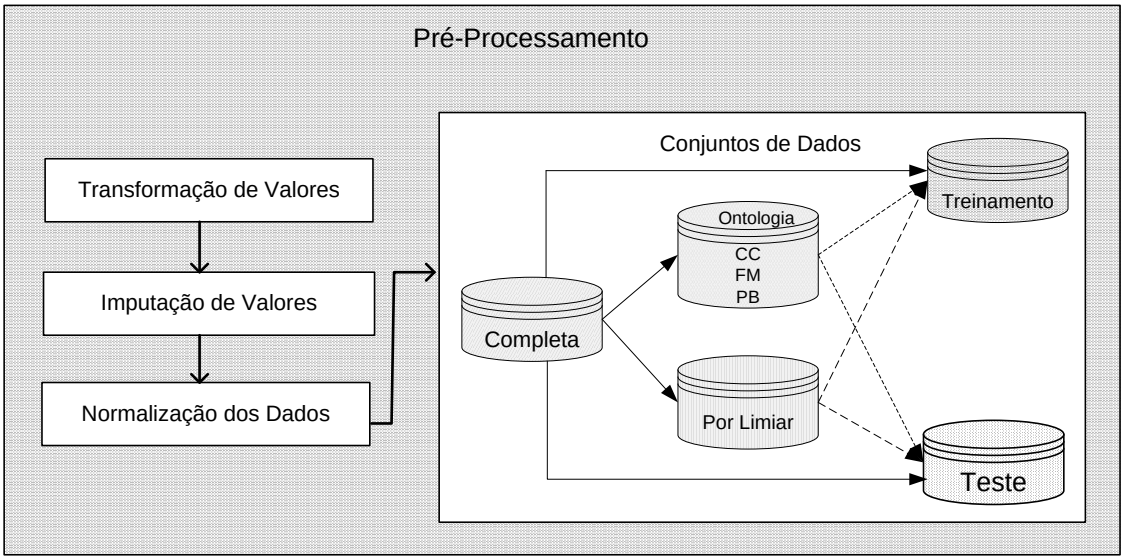


Figura 36 – Etapa de Pré-Processamento.

Transformação dos Valores

Esse passo consiste na transformação dos valores dos atributos das bases de dados de categóricos para valores contínuos. Isto é necessário porque redes neurais não trabalham com dados categóricos. Essa transformação foi feita atribuindo valores sequências (1, 2, 3,...) para os atributos categóricos.

Imputação dos Valores

Várias amostras dos conjuntos de dados possuem valores faltantes. Sendo assim, foi necessária a imputação desses valores.

A imputação de valores pode permitir uma melhor eficiência do algoritmo. Porém, uma imputação errada pode gerar mais problemas do que os resolver. Para isso, existem várias técnicas que podem ser usadas nesse processo, tais como: substituir os valores faltantes por uma constante, substituir os valores faltantes pela média, pelo valor mais comum, entre vários outros.

O critério usado para imputar os valores ausentes foi calcular a média aritmética de todos os ancestrais mais próximos da classe à qual pertence a amostra. Nas amostras multirrótulos é feita a média aritmética também sobre a quantidade de rótulos da amostra.

Normalização dos Dados

Após a transformação e a imputação dos valores foi feita a normalização desses dados. A técnica utilizada foi a Min-Max, Equação 50. Essa técnica faz a transformação linear dos valores v de um atributo a de modo a mapear os valores mínimo e máximo originais, $\min_a$ e $\max_a$ para valores mínimo e máximo da nova escala.

$$v' = \frac{v - \min_a}{\max_a - \min_a}$$ **Equação 50**

Conjuntos de Dados

Vários experimentos foram feitos com diferentes formatações das bases de dados. Todas as bases de dados foram divididas entre conjunto de treinamento e teste. Para isso dividiu-se todas as bases de dados criadas no passo anterior em 2/3 do conjunto para o treinamento do algoritmo e 1/3 para o teste.

- **Base de Dados Completa**

Essa formatação refere-se as bases de dados com originais, apenas com os valores faltantes imputados. Vale ressaltar que a hierarquia com todas as classes foram usadas.

- **Base de Dados Separadas por Ontologias**

Nesse tipo de formatação, para cada um dos dez conjuntos de dados foram divididos nas três ontologias: componente celular, função molecular e processo biológico. A Tabela 10 mostra as características das bases de dados após essa divisão.

- **Base de Dados com Exemplos Removidos conforme um Limiar**

Foi feita uma análise da quantidade de exemplos que cada classe da hierarquia possui, visto que se uma determinada classe tiver poucos exemplos esses poderão atrapalhar o processo de aprendizagem do algoritmo. Observou-se que nas bases de dados havia classes com poucas quantidades de instâncias. Sendo assim, foram estabelecidos limiares referentes à quantidade mínima de instâncias que cada classe deveria ter levando em consideração também as suas classes ancestrais. A partir desses limiares foram removidos classes dos exemplos que estivesse abaixo desses valores.

Tabela 10 – Características das Bases de Dados Separadas por Ontologias.

Bases de Dados	Ontologias	Quant. Amostras	Quant. Classes	Quant. Max. Níveis	Quant. Min/Max Classes por Amostras	Quant. Min/Max Amostras por Classes
Cellcycle	CC	3751	547	9	1/9	0/785
	FM	3751	1544	13	1/8	0/603
	PB	3751	2034	13	1/20	0/263
Church	CC	3749	547	9	1/9	0/786
	FM	3749	1544	13	1/8	0/602
	PB	3749	2034	13	1/20	0/264
Derisi	CC	3719	547	9	1/9	0/781
	FM	3719	1539	13	1/8	0/591
	PB	3719	2033	13	1/20	0/259
Eisen	CC	2418	517	9	1/9	0/492
	FM	2418	1283	13	1/8	0/220
	PB	2418	1773	13	1/20	0/212
Expr	CC	3773	547	9	1/9	0/789
	FM	3344	1546	13	1/8	0/608
	PB	3773	2038	13	1/20	0/264
Gasch1	CC	93	547	9	1/2	0/786
	FM	3758	1544	13	1/8	0/604
	PB	3758	2034	13	1/20	0/264
Gasch2	CC	3773	547	9	1/9	0/789
	FM	3773	1544	13	1/8	0/608
	PB	3773	2038	13	1/20	0/264
Pheno	CC	1586	462	9	1/7	0/388
	FM	1586	1047	13	1/5	0/258
	PB	1586	1618	12	1/14	0/112
Seq	CC	3900	547	9	1/9	0/791
	FM	3900	1546	13	1/8	0/615
	PB	3900	2040	13	1/20	0/269
Spo	CC	3697	547	9	1/9	0/775
	FM	3697	1539	13	1/8	0/590
	PB	3697	2033	13	1/20	0/258

Fizeram-se experimentos com vários limiares tais como 150, 200, 300 entre outros. Após uma análise das características desses novos conjuntos, observou que com limiares iguais a 150 e a 200 havia pouca diferença na característica dos conjuntos. Dessa maneira, optou-se por fazer o treinamento nos conjuntos de dados com o limiar igual a 300.

A Tabela 11 mostra a características desses novos conjuntos de dados que tiveram alterações após esse pré-processamento. Observa-se que houve uma diminuição na quantidade de amostras nas Bases de dados Eisen e Pheno. Além disso, em todos os conjuntos de dados houve uma grande redução na quantidade mínima e máxima de classes por amostra. As outras características que foram apresentadas na Tabela 8 não serão mostradas nesta tabela porque os dados permaneceram os mesmos. Essas bases estão disponíveis http://www.ppgia.pucpr.br/~nievola/BasesDadosHelyane/.

Tabela 11 – Características das Bases de Dados GO conforme o Limiar.

Base de Dados	Quant. Amostras	Quant. Min/Max Classes por Amostras
Cellcycle (SPELLMAN et al., 1998)	3751	1/15
Church (ROTH et al., 1998)	3749	1/15
Derisi (DERISI et al, 1997)	3719	1/13
Eisen (EISEN et al., 1998)	1485	1/4
Expr (CLARE, 2003)	3773	1/15
Gasch1 (GASCH et al., 2000)	3758	1/15
Gasch2 (GASCH et al., 2001)	3773	1/15
Pheno (CLARE, 2003)	984	1/4
Seq (CLARE, 2003)	3900	1/16
Spo (CHU et al., 1998)	3697	1/13

6.3.2 Classificação Hierárquica

Os algoritmos de classificação hierárquica que foram selecionados para a realização dos experimentos estão citados na Figura 37. Todos os conjuntos de dados foram submetidos as esses classificadores. Os parâmetros do MHC-CNN foram a taxa de aprendizagem inicial (μ_i) igual 0,1, a taxa de aprendizagem final (μ_f) igual a 0,001 e a constante *C* igual a 100. Já para o MHC-ES foi estabelecida uma população (P) com 50 indivíduos e a constante c com valor igual a 0,7. Quanto aos classificadores Clus-HSC e Clus-HMC foram utilizados os parâmetros padrões dos algoritmos.

6.3.3 Avaliação Estatística

A avaliação dos resultados obtidos pelos classificadores foi feita utilizando o teste de hipótese de Friedman (FRIEDMAN, 1937, 1940). Esse teste é indicado para comparar o desempenho de vários algoritmos em diferentes bases de dados. O teste é sugerido por ser não paramétrico, haja vista a dificuldade de se conhecer a distribuição dos dados.

Segundo os autores Iman e Davenport (1980) o teste de Friedman é considerado muito conservador, e por tal motivo, sugere o uso da estatística FF distribuída de acordo com a distribuição F de Snedecor.

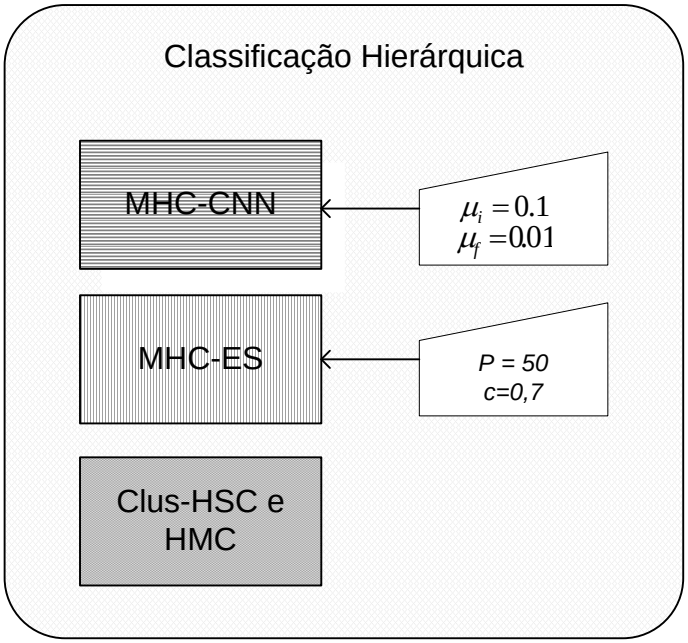


Figura 37 – Etapa de Classificação Hierárquica

Para a análise de quais algoritmos têm diferença significativa de desempenho, Demsar (2006) sugere o teste de Nemenyi, que calcula a distância crítica entre os desempenhos, o qual foi usado nestas situações dentro desta tese.

6.4 RESULTADOS EXPERIMENTAIS

Esta Seção será dividida conforme foi feito o processamento das bases de dados, ou seja, serão apresentados os resultados obtidos com a base de dados completa, a base separada nas três ontologias e por fim as bases com as instâncias removidas conforme o limiar estabelecido.

6.4.1 Resultados Bases de Dados Completa

Nesta Seção serão apresentados os resultados obtidos pelos classificadores MHC-CNN e MHC-ES. A Figura 38 apresenta os resultados obtidos com a execução do MHC-CNN, com 50, 500 e 1000 épocas, nas dez bases de dados usando duas medidas de avaliação: distância e Medida-Fh, conforme citadas nas seções 3.4.1.2 e 3.4.3, respectivamente. Observa-se um resultado maior usando a medida de distância em todas as bases de dados. Uma possível causa da medida de distância apresentar resultados melhores deve-se ao grande número de predições ocorridas em níveis mais próximos da raiz.

A Figura 39 apresenta os resultados obtidos com o algoritmo MHC-ES. Nota-se que a medida de distância também apresentou resultados superiores que a medida-Fh. Embora tenham sido feito experimentos com diversos valores para a geração de indivíduos, apenas serão apresentados resultados das execuções com 20, 60 e 100 gerações.

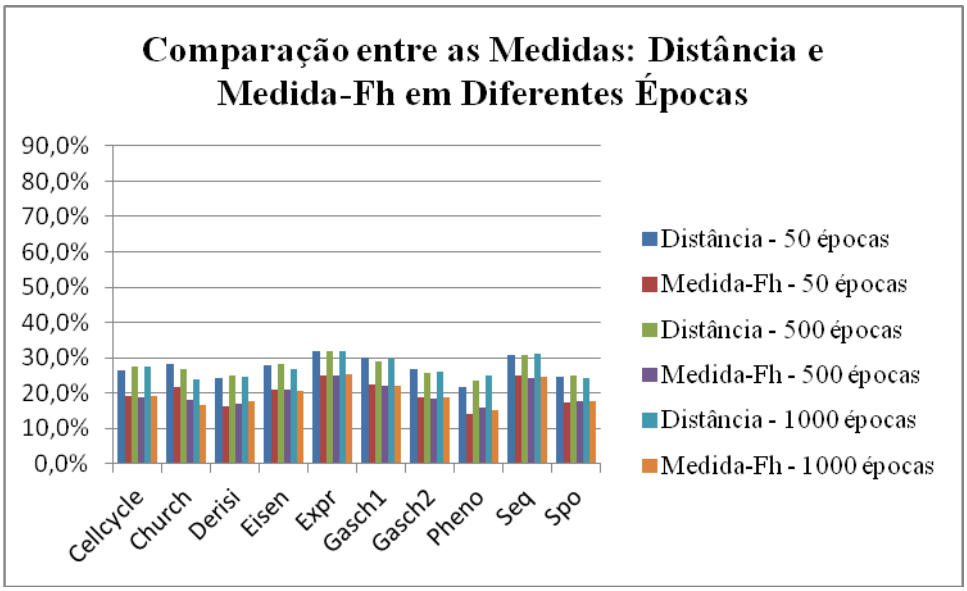


Figura 38 – Resultados do MHC-CNN na Base de Dados Completa.

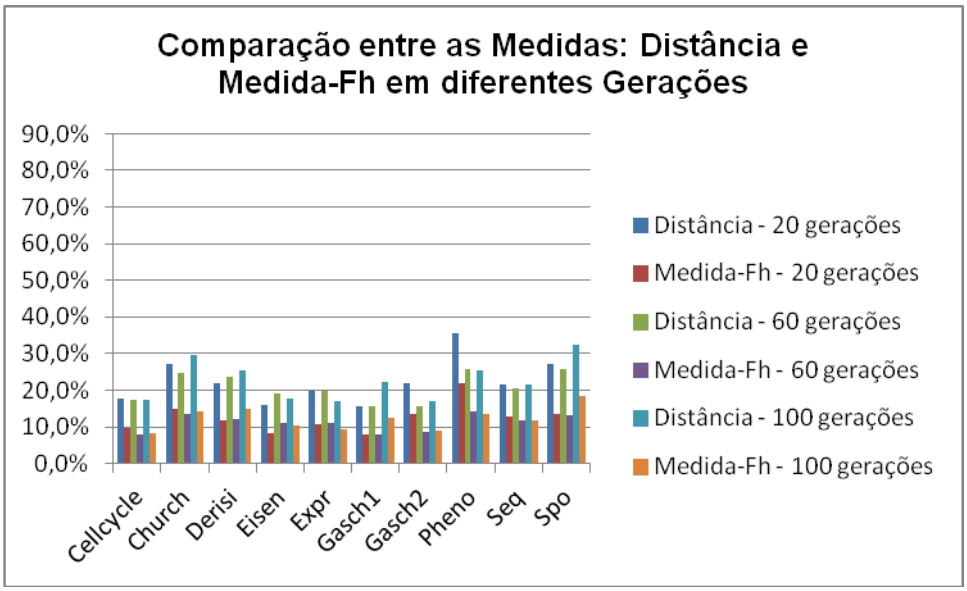


Figura 39 - Resultados do MHC-ES na Base de Dados Completa.

6.4.2 Resultados Bases de Dados com Exemplos Removidos conforme Limiar

A Figura 40 apresenta o resultado obtido com o MHC-CNN em três diferentes épocas nas bases de dados com Limiar igual a 300.

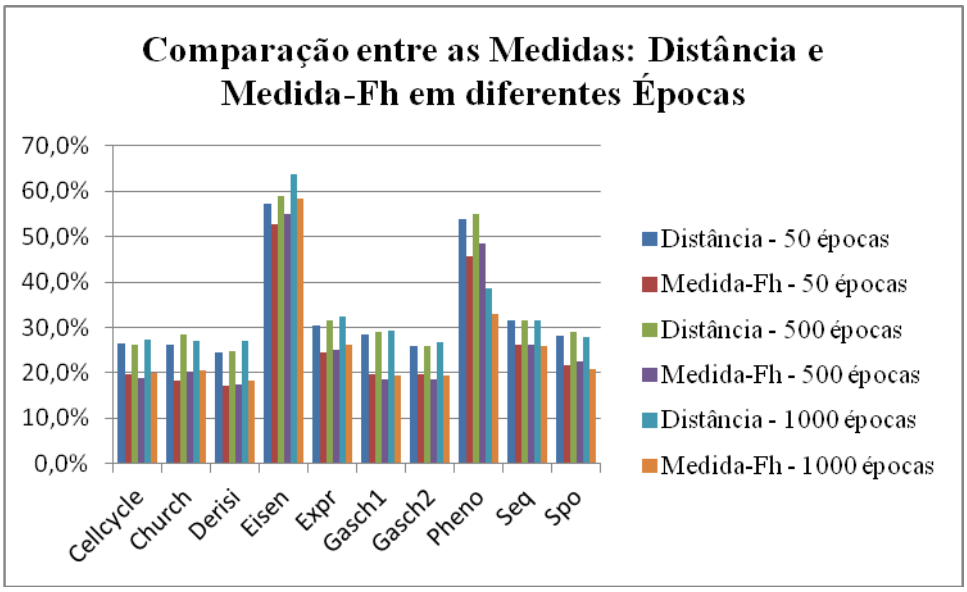


Figura 40 – Resultado do MHC-CNN na Base de Dados com Limiar 300.

A Figura 41 apresenta o resultado obtido com o MHC-ES nas bases de dados com Limiar igual a 300 com 20, 60 e 100 gerações.

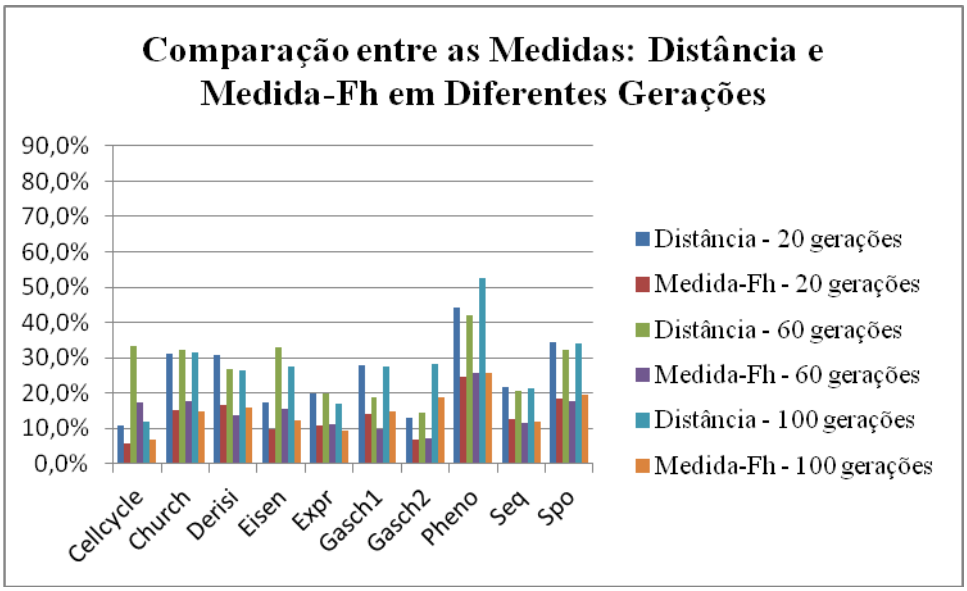


Figura 41 - Resultado do MHC-ES na Base de Dados com Limiar 300.

6.4.3 Resultados Bases de Dados Separadas por Ontologias

Nesta Seção serão apresentados os resultados obtidos pelos classificadores MHC-CNN, MHC-ES, Clus-HMC e Clus-HSC nas três ontologias. Apesar de terem sido feitos diversos experimentos serão mostrados apenas os resultados obtidos com a execução de 50, 500 e 1000 épocas.

A Figura 42 apresenta os resultados obtidos na base de dados da ontologia Componente Celular. Nota-se que a medida de distância obteve uma taxa de acerto maior que a medida-Fh em ambas as épocas.

A Figura 43 mostra os resultados obtidos na base de dados da ontologia Função Molecular. Observa-se que as taxas de acerto de ambas as medidas foram inferiores do resultado comentado anteriormente. O mesmo ocorreu nos resultados obtidos na base de dados da ontologia Processo Biológico, conforme pode ser visualizado na Figura 44.

Uma das causas dessa variação entre as bases de dados das três ontologias deve-se à quantidade de classes. Observa-se que todas as dez bases de dados pertencentes à ontologia Componente Celular possuem uma quantidade de classes inferior às demais ontologias. Além disso, a quantidade máxima de classes que uma instância possui nessa ontologia são nove classes. As duas observações feitas facilitam o treinamento do modelo fazendo com que o classificador consiga predizer melhor os dados.

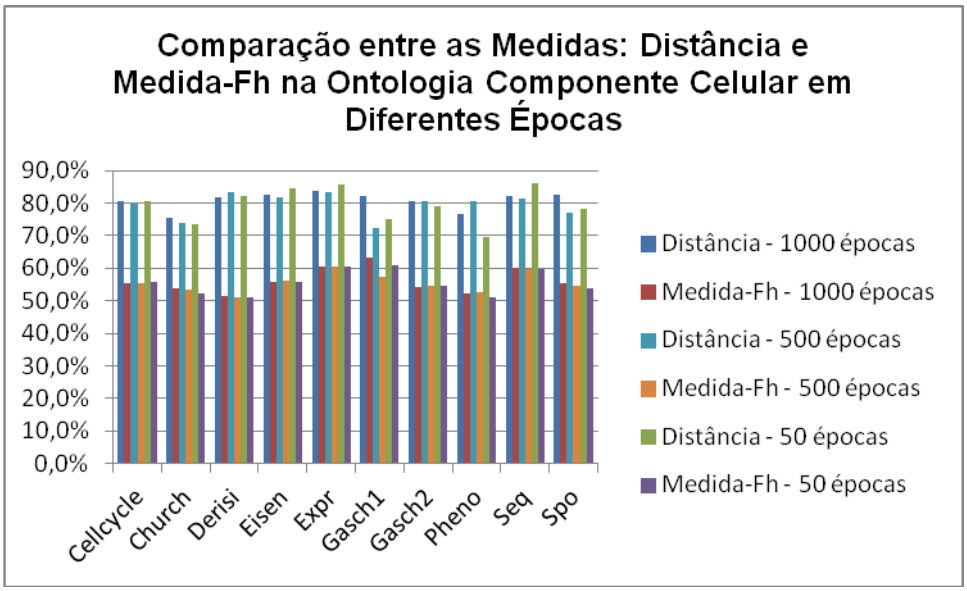


Figura 42 – Resultados do MHC-CNN na Ontologia Componente Celular.

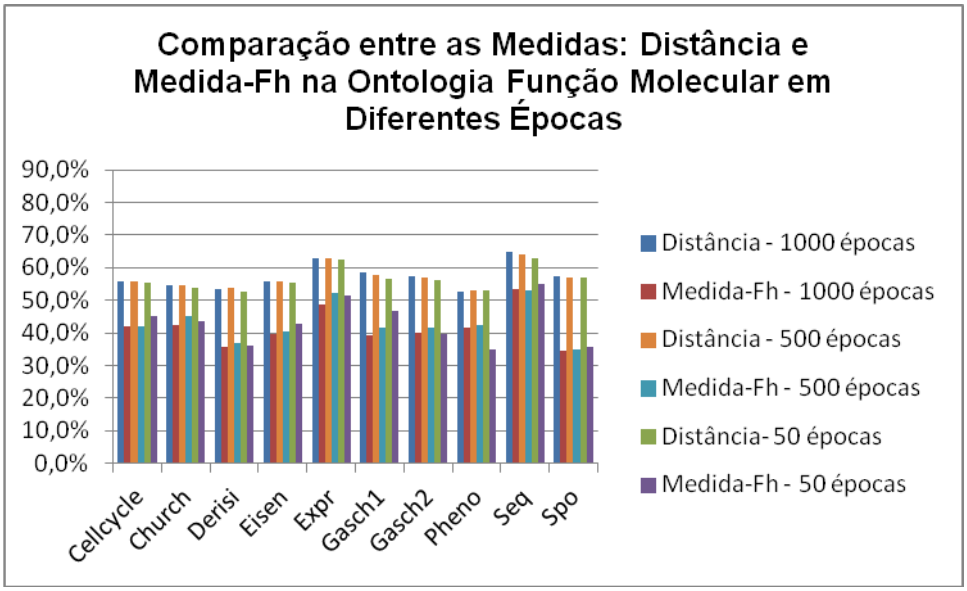


Figura 43 – Resultados do MHC-CNN na Ontologia Função Molecular.

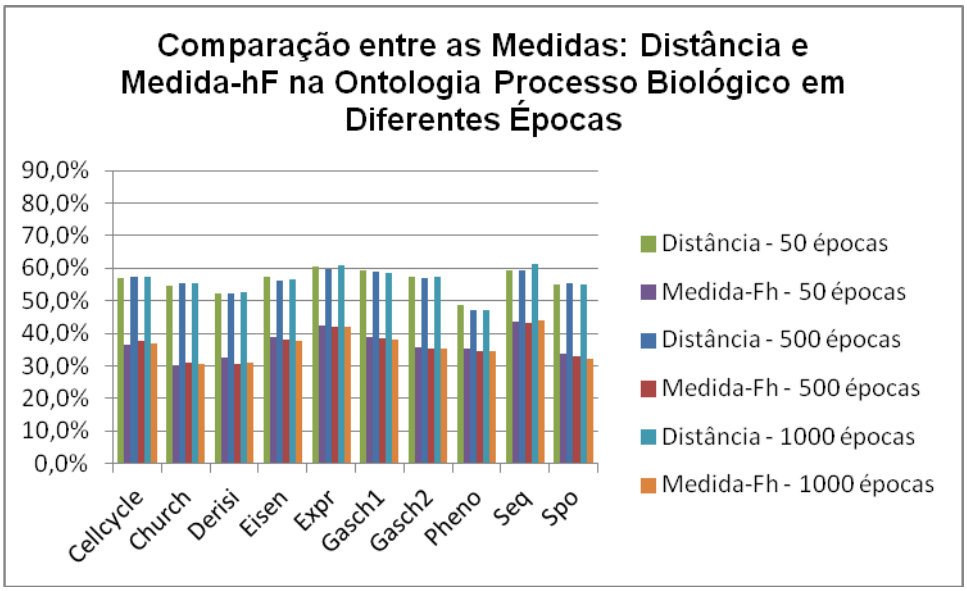


Figura 44 – Resultados do MHC-CNN na Ontologia Processo Biológico.

Para a execução de experimentos com o algoritmo MHC-ES foram selecionandos vários valores para as gerações. Aqui serão apresentados os resultados de três diferentes gerações, tais quais iguais a 20, 60 e 100.

A Figura 45 apresenta o resultado obtido na base de dados da ontologia Componente Celular. Observa-se que os resultados obtidos foram próximos aos obtidos pelo MHC-CNN.

A Figura 46 mostra os resultados obtidos na base de dados da ontologia Função Molecular. A Figura 47 mostra os resultados obtidos na base de dados da ontologia Processo Biológico.

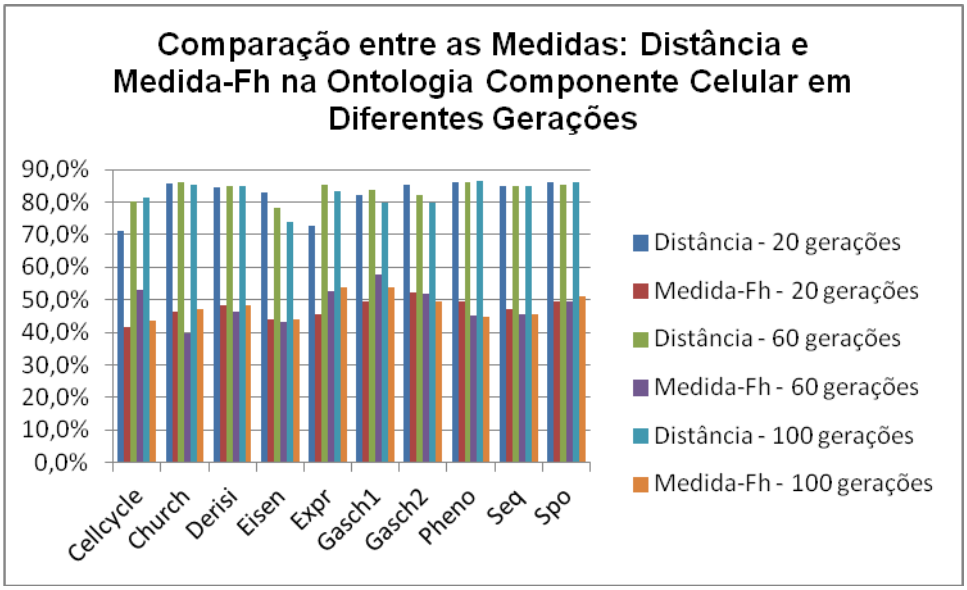


Figura 45 - Resultados do MHC-ES na Ontologia Componente Celular.

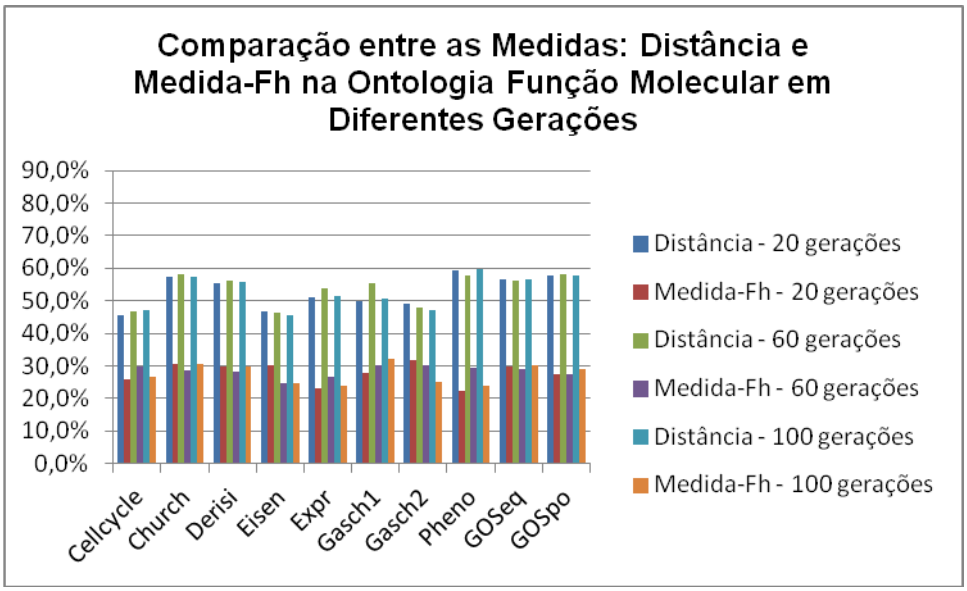


Figura 46 - Resultados do MHC-ES na Ontologia Função Molecular.

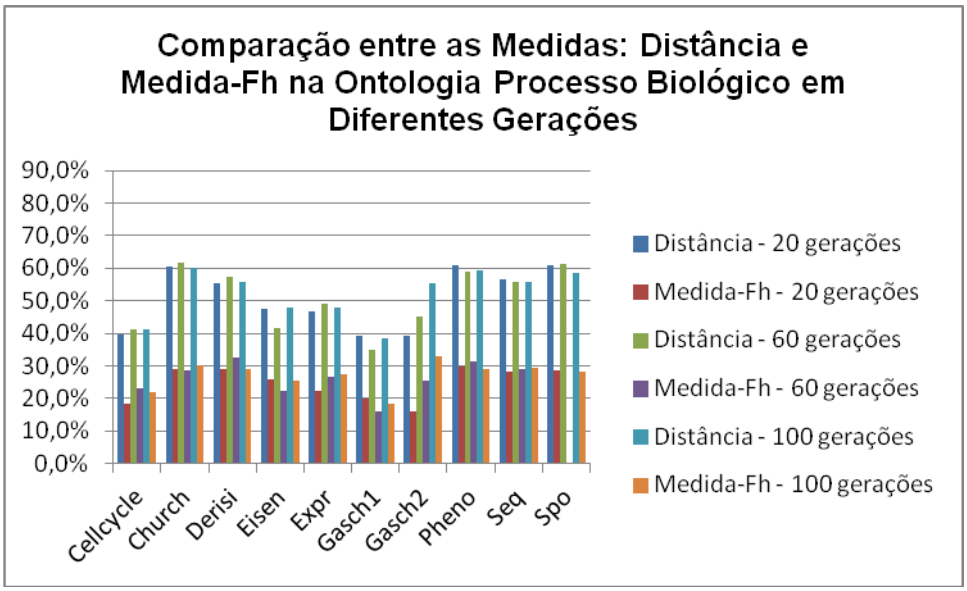


Figura 47 - Resultados do MHC-ES na Ontologia Processo Biológico.

6.5 COMPARAÇÃO DOS RESULTADOS

Nesta Seção serão apresentadas as comparações dos resultados obtidos pelos classificadores desenvolvidos: MHC-CNN e MHC-ES e os classificadores desenvolvidos por Vens et. al (2008): Clus-HMC e Clus-HSC.

6.5.1 Comparativo entre MHC-CNN e MHC-ES

Nesta seção serão apresentados os comparativos feitos entre os algoritmos MHC-CNN e MHC-ES utilizando o teste de Friedman para avaliar estatisticamente os resultados obtidos através da medida de distância e da medida-Fh da base de dados completa, da base de dados com limiar 300 e das bases separadas por ontologias.

6.5.1.1 Base de Dados Completa

A Tabela 12 mostra os resultados dos dois algoritmos, já ilustrados na Seção 6.4.1 quando utilizado a medida de distância. Após o cálculo dos postos médio e aplicando nesses dados a Equação 40, têm-se $X_F^2 = 6{,}96$. Em seguida, usando a estatística F_F para a correção de X_F^2, conforme a Equação 41, tem-se como resultado $F_F = 1{,}5$.

Para os graus de liberdade $k-1$ e $(k-1)*(n-1)$ em que $k=10$ e $n=6$, encontra-se o seguinte valor tabelado na distribuição F de Snedecor tem-se $F(9;45) = 2{,}09$. Assim, $F_F < F(9;45)$ logo a hipótese nula não pode ser rejeitada, indicando que não há diferença estatística entre os resultados dos algoritmos comparados.

Tabela 12 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida de distância.

	MHC-CNN (em épocas)			**MHC-ES (em gerações)**		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	26,5%	27,3%	27,5%	17,8%	17,2%	17,3%
Church	28,2%	26,8%	23,7%	27,1%	24,8%	29,5%
Derisi	24,3%	25,1%	24,5%	22,0%	23,6%	25,5%
Eisen	27,8%	28,1%	26,8%	16,0%	19,1%	17,6%
Expr	31,9%	31,9%	31,8%	19,8%	19,9%	17,0%
Gasch1	30,0%	29,0%	29,6%	15,5%	15,7%	22,2%
Gasch2	26,6%	25,6%	26,1%	22,1%	15,8%	17,1%
Pheno	21,5%	23,7%	24,9%	35,6%	25,9%	25,4%
Seq	21,5%	30,8%	31,2%	21,6%	20,7%	21,5%
Spo	24,6%	24,8%	24,4%	27,1%	25,8%	32,5%

Os resultado referentes a medida-Fh, em ambos os algoritmos, também já ilustrados na Seção 6.4.1, são mostrados na Tabela 13. Após o cálculo dos postos médio têm-se $X_F^2 = 28{,}7$ e $F_F = 12{,}1$.

Para os mesmos graus de liberdade tem-se $F(9;45) = 2,09$ do resultado anterior determina-se que a hipótese nula pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Tabela 13 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida-Fh.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	19,0%	18,8%	19,2%	9,6%	7,9%	8,2%
Church	21,6%	18,2%	16,6%	14,9%	13,6%	14,1%
Derisi	16,1%	17,0%	17,5%	11,8%	12,2%	14,9%
Eisen	21,0%	21,0%	20,8%	8,1%	10,9%	10,3%
Expr	24,8%	25,0%	25,2%	10,8%	11,1%	9,3%
Gasch1	22,4%	21,9%	22,1%	7,9%	8,0%	12,4%
Gasch2	18,8%	18,3%	18,6%	13,4%	8,7%	8,8%
Pheno	13,9%	15,7%	15,2%	21,7%	14,2%	13,7%
Seq	13,9%	24,3%	24,6%	12,7%	11,6%	12,0%
Spo	17,2%	17,6%	17,6%	13,6%	13,3%	18,4%

Rejeitada a hipótese nula, deve-se identificar quais algoritmos possuem diferenças significativas de desempenho considerando a distância crítica (CD), calculada através da Equação 42.

Considerando o nível de significância de 95% tem-se o valor de $CD = 12,1$. Considerando os dados da Tabela 13, o algoritmo MHC-CNN, nos três casos selecionados (50, 500 e 1000 épocas) é estatisticamente superior aos resultados do algoritmo MHC-ES apenas com a seleção de 20 e 60 gerações.

6.5.1.2 Base de Dados com Limiar igual a 300

A Tabela 14 apresenta um comparativo entre os algoritmos MHC-CNN e MHC-ES quando aplicado nas bases de dados com limiar igual a 300.

Analisando estatisticamente através do Teste de Friedman e aplicando nesses dados a Equação 40, têm-se $X_F^2 = 3,53$, $F_F = 0,7$ e $F(9;45) = 2,09$ graus de liberdade. Como, $F_F < F(9;45)$ a hipótese nula não pode ser rejeitada, indicando que não há diferença estatística entre os resultados dos algoritmos comparados.

Tabela 14 - Comparativo entre o MHC-CNN e o MHC-ES usando a medida de distância.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	26,4%	26,1%	27,2%	10,9%	33,3%	11,7%
Church	26,0%	28,5%	27,2%	31,1%	32,3%	31,4%
Derisi	24,4%	24,8%	26,9%	30,8%	26,7%	26,4%
Eisen	57,3%	59,1%	63,9%	17,3%	32,8%	27,6%
Expr	30,5%	31,6%	32,5%	19,8%	19,9%	17,0%
Gasch1	28,6%	29,1%	29,2%	28,0%	18,6%	27,4%
Gasch2	25,9%	25,8%	26,6%	12,8%	14,4%	28,1%
Pheno	53,9%	55,1%	38,6%	44,2%	41,9%	52,7%
Seq	31,5%	31,6%	31,5%	21,6%	20,7%	21,5%
Spo	28,2%	28,9%	28,0%	34,5%	32,3%	34,0%

Realizando o mesmo procedimento dos resultados obtidos pela medida-Fh, em ambos os algoritmos, tem-se a comparação ilustrada na Tabela 15.

Nesse caso observa-se que existe diferença estatística entre os resultados dos dois algoritmos. Para os mesmos graus de liberdade tem-se $F(9;45) = 2{,}09$, $X_F^2 = 38{,}2$ e $F_F = 29{,}0$. Assim, $F_F > F(9;45)$ indicando que a hipótese nula deve ser rejeitada.

Tabela 15 - Comparativo entre o MHC-CNN e o MHC-ES usando a medida-Fh.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	19,8%	19,0%	19,9%	5,8%	17,2%	6,6%
Church	18,3%	20,2%	20,6%	15,3%	17,8%	14,9%
Derisi	17,1%	17,5%	18,4%	16,6%	13,6%	15,9%
Eisen	52,8%	55,1%	58,4%	9,6%	15,3%	12,1%
Expr	24,6%	25,1%	26,1%	10,8%	11,1%	9,3%
Gasch1	19,7%	18,6%	19,3%	14,0%	9,7%	14,9%
Gasch2	19,6%	18,6%	19,3%	6,8%	7,2%	18,6%
Pheno	45,6%	48,6%	32,9%	24,5%	25,8%	25,8%
Seq	26,2%	26,2%	25,9%	12,7%	11,6%	11,7%
Spo	21,6%	22,5%	20,7%	18,3%	17,6%	19,5%

Rejeitada a hipótese nula, identificam-se quais algoritmos possuem diferenças significativas de desempenho considerando a distância crítica (CD) cujo resultado é $CD = 2{,}38$, considerando o nível de significância de 95%. Considerando os resultados apresentados nota-se que o algoritmo MHC-CNN (nas três épocas selecionadas) é superior ao algoritmo MHC-ES em todas as gerações.

6.5.1.3 Bases de Dados Separadas por Ontologias

Como sabe-se as base de dados GO são formadas por três ontologias e é dessa maneira que os resultados obtidos serão mostrados. Nesta Seção serão apresentados os resultados obtidos da ontologia Componente Celular. Os resultados das duas outras ontologias: Função Molecular e Processo Biológico serão mostrados no Apêndice B.

A Tabela 16 mostra os resultados dos dois algoritmos, já ilustrados na Seção 6.4.3 quando utilizada a medida de distância.

Recalculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 10{,}66$ e $F_F = 2{,}4$. Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2{,}09$. Deste modo a hipótese nula é rejeitada, $F_F > F(9;45)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Tabela 16 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida de distância na Ontologia Componente Celular.

	MHC-CNN (em épocas)			**MHC-ES (em gerações)**		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	80,5%	79,8%	80,6%	71,3%	80,0%	81,4%
Church	73,7%	73,8%	73,8%	85,6%	86,2%	85,4%
Derisi	82,4%	83,5%	81,7%	84,6%	85,1%	84,8%
Eisen	84,7%	82,0%	82,5%	82,8%	78,3%	73,8%
Expr	85,7%	83,4%	83,7%	72,9%	85,5%	83,5%
Gasch1	74,9%	72,5%	82,3%	82,2%	83,6%	79,9%
Gasch2	79,0%	80,5%	80,6%	85,4%	82,4%	86,7%
Pheno	69,5%	80,7%	76,5%	86,2%	86,0%	84,9%
Seq	85,9%	81,3%	82,3%	85,0%	85,0%	84,9%
Spo	78,3%	77,1%	82,5%	86,0%	85,5%	58,6%

Rejeitada a hipótese nula, identificam-se quais algoritmos possuem diferenças significativas de desempenho considerando a distância crítica (CD), calculada através da Equação 42.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}38$. Considerando os dados da Tabela 16, o algoritmo MHC-ES, com 60 gerações, é estatisticamente superior aos resultados do algoritmo MHC-CNN com a execução de 500 épocas.

A Tabela 17 mostra os resultados dos dois algoritmos, também já ilustrados na Seção 7.3.3 quando utilizado a medida-Fh.

Calculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 37{,}9$ e $F_F = 28{,}3$. Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2{,}09$. Deste modo a $F_F > F(9;45)$ logo a hipótese nula pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}38$. Considerando os dados da Tabela 17, o algoritmo MHC-CNN, nos três casos selecionados (50, 500 e 1000 épocas) é estatisticamente superior aos resultados do algoritmo MHC-ES também nos três casos selecionados: 20, 60 e 100 gerações.

Tabela 17 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida-Fh na Ontologia Componente Celular.

	MHC-CNN (em épocas)			**MHC-ES (em gerações)**		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	55,8%	55,3%	55,4%	41,5%	53,0%	43,6%
Church	52,3%	53,6%	53,7%	46,2%	39,6%	47,0%
Derisi	50,9%	51,1%	51,3%	48,4%	46,5%	48,2%
Eisen	50,9%	56,1%	55,6%	44,1%	43,2%	44,1%
Expr	60,4%	60,4%	60,4%	45,7%	52,5%	53,9%
Gasch1	60,9%	57,2%	63,3%	49,5%	57,6%	53,8%
Gasch2	54,8%	54,6%	54,2%	52,1%	51,9%	49,4%
Pheno	51,0%	52,8%	52,2%	49,6%	45,2%	44,9%
Seq	59,8%	59,6%	60,1%	47,2%	45,6%	45,4%
Spo	53,7%	54,8%	55,3%	49,7%	49,3%	28,2%

6.5.2 Comparativo entre MHC-CNN e o Clus

Devido a medida de avaliação AUPRC que o algoritmo Clus-HMC e Clus-HSC utiliza, o algoritmo MHC-CNN teve que ser avaliado da mesma maneira para que fosse possível a comparação entre os algoritmos. Dessa maneira, avaliou-se a medida AUPRC com diferentes épocas. Por questões de espaço são mostrados os resultados obtidos com 50, 500 e 1000 épocas.

6.5.2.1 Base de Dados Completa

A Tabela 18 apresenta os resultados apresentados na base de dados completa. Analisando estatisticamente esses resultados através do teste de Friedman têm-se $X_F^2 = 32{,}56$ e $F_F = 39{,}4$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Rejeitada a hipótese nula e considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC e Clus-HSC foi estatisticamente superior ao MHC-CNN com 50 e 500 épocas. Já o resultado do MHC-CNN com 1000 épocas foi estatisticamente inferior apenas ao Clus-HMC.

Tabela 18 – Comparativo da Medida AUPRC entre o MHC-CNN e o Clus-HMC e Clus-HSC na Base de Dados Completa.

	MHC-CNN (em épocas)			**Clus**	
	50	**500**	**1000**	**HMC**	**HSC**
Cellcycle	0,09	0,09	0,10	0,44	0,32
Church	0,13	0,13	0,12	0,45	0,38
Derisi	0,12	0,12	0,14	0,44	0,35
Eisen	0,09	0,11	0,11	0,46	0,35
Expr	0,15	0,13	0,14	0,47	0,28
Gasch1	0,11	0,08	0,08	0,46	0,31
Gasch2	0,10	0,11	0,11	0,44	0,34
Pheno	0,12	0,10	0,10	0,42	0,39
Seq	0,09	0,09	0,10	0,47	0,28
Spo	0,09	0,11	0,12	0,47	0,36

Também, avaliaram-se as duas medidas que dão origem à medida AUPRC: precisão e revocação nas 10 bases de dados.

Para fins de demonstração, selecionaram-se duas bases de dados para apresentar os resultados obtidos com 50, 500 e 1000 épocas. As bases de dados selecionadas foram a Church e Derisi. A primeira foi selecionada por ser uma das bases que possui atributos faltantes, os quais foram imputados na etapa pré-processamento. Já a segunda base de dados nenhuma alteração foi feita, pois essa possuía todos os dados.

A

Tabela 19 apresenta os resultados obtidos nas duas bases de dados com a execução de 50 épocas. Observa-se que em todos os limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. O mesmo ocorre na execução com 500 épocas conforme mostra a Tabela 20.

Tabela 19 – Comparativo da Medida de Precisão e Revocação com 50 épocas na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**0,89**	0,32	0,55	0,34	0,46	0,01	**0,86**	0,32	0,54	0,30	0,44
T(20)	0,01	**0,86**	0,49	0,44	0,49	0,38	0,01	**0,83**	0,49	0,43	0,43	0,37
T(30)	0,01	**0,84**	0,59	0,38	0,60	0,31	0,02	**0,82**	0,61	0,36	0,53	0,30
T(40)	0,01	**0,82**	0,67	0,33	0,67	0,27	0,02	**0,81**	0,67	0,31	0,62	0,27
T(50)	0,02	**0,80**	0,80	0,25	0,75	0,23	0,02	**0,79**	0,79	0,24	0,70	0,23
T(60)	0,02	**0,75**	0,89	0,20	0,83	0,20	0,02	**0,77**	0,86	0,20	0,78	0,20
T(70)	0,05	**0,60**	0,92	0,19	0,86	0,19	0,03	**0,73**	0,92	0,17	0,83	0,18
T(80)	0,15	**0,40**	0,94	0,17	0,89	0,16	0,07	**0,53**	0,93	0,16	0,87	0,17
T(90)	0,23	**0,19**	0,98	0,13	0,93	0,14	0,39	**0,14**	0,97	0,12	0,92	0,14
T(100)	0,36	**0,10**	0,99	0,10	0,95	0,10	0,40	**0,14**	1,00	0,09	0,96	0,09

Tabela 20 – Comparativo da Medida de Precisão e Revocação com 500 épocas na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0,0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10,0)	0,01	**0,88**	0,32	0,55	0,34	0,46	0,02	**0,81**	0,32	0,54	0,30	0,44
T(20,0)	0,01	**0,84**	0,49	0,44	0,49	0,38	0,02	**0,79**	0,49	0,43	0,43	0,37
T(30,0)	0,01	**0,82**	0,59	0,38	0,60	0,31	0,02	**0,78**	0,61	0,36	0,53	0,30
T(40,0)	0,02	**0,81**	0,67	0,33	0,67	0,27	0,02	**0,78**	0,67	0,31	0,62	0,27
T(50,0)	0,02	**0,79**	0,80	0,25	0,75	0,23	0,02	**0,76**	0,79	0,24	0,70	0,23
T(60,0)	0,03	**0,73**	0,89	0,20	0,83	0,20	0,04	**0,69**	0,86	0,20	0,78	0,20
T(70,0)	0,04	**0,67**	0,92	0,19	0,86	0,19	0,08	**0,53**	0,92	0,17	0,83	0,18
T(80,0)	0,06	**0,62**	0,94	0,17	0,89	0,16	0,08	**0,50**	0,93	0,16	0,87	0,17
T(90,0)	0,09	**0,53**	0,98	0,13	0,93	0,14	0,12	**0,44**	0,97	0,12	0,92	0,14
T(100,0)	0,32	**0,11**	0,99	0,10	0,95	0,10	0,31	**0,11**	1,00	0,09	0,96	0,09

Já na execução com 1000 épocas apenas no limiar T(100) a medida foi inferior aos outros dois algoritmos conforme mostra a Tabela 21.

Tabela 21 – Comparativo da Medida de Precisão e Revocação com 1000 épocas na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**0,87**	0,32	0,55	0,34	0,46	0,02	**0,81**	0,32	0,54	0,30	0,44
T(20)	0,01	**0,84**	0,49	0,44	0,49	0,38	0,02	**0,79**	0,49	0,43	0,43	0,37
T(30)	0,01	**0,82**	0,59	0,38	0,60	0,31	0,02	**0,78**	0,61	0,36	0,53	0,30
T(40)	0,02	**0,81**	0,67	0,33	0,67	0,27	0,02	**0,78**	0,67	0,31	0,62	0,27
T(50)	0,02	**0,79**	0,80	0,25	0,75	0,23	0,02	**0,77**	0,79	0,24	0,70	0,23
T(60)	0,03	**0,74**	0,89	0,20	0,83	0,20	0,04	**0,69**	0,86	0,20	0,78	0,20
T(70)	0,04	**0,70**	0,92	0,19	0,86	0,19	0,07	**0,55**	0,92	0,17	0,83	0,18
T(80)	0,05	**0,63**	0,94	0,17	0,89	0,16	0,08	**0,53**	0,93	0,16	0,87	0,17
T(90)	0,08	**0,56**	0,98	0,13	0,93	0,14	0,10	**0,46**	0,97	0,12	0,92	0,14
T(100)	0,34	0,08	0,99	**0,10**	0,95	**0,10**	0,33	**0,13**	1,00	0,09	0,96	0,09

Com objetivo de comparar os resultados obtidos e mostrados nas três tabelas anteriores, a Figura 48 apresenta os resultados da base de dados Church e a Figura 49 os resultados obtidos da base de dados Derisi.

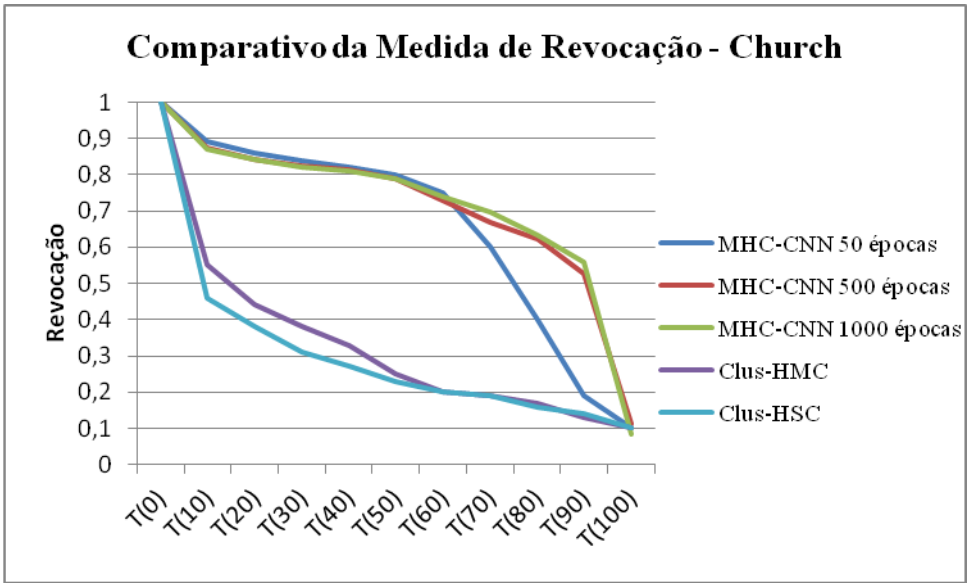


Figura 48 – Comparativo Geral da Medida de Revocação na Base de Dados Completa Church.

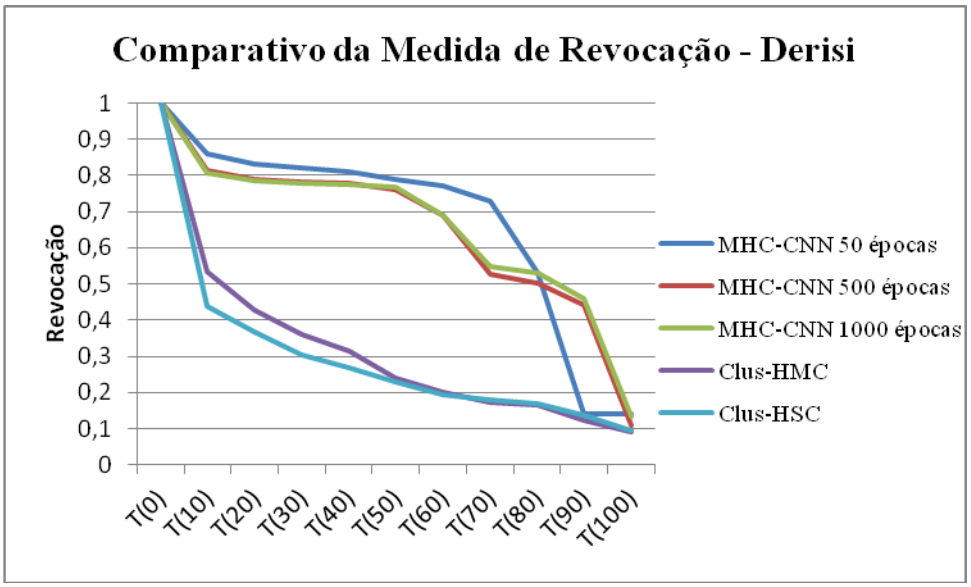


Figura 49 – Comparativo Geral da Medida de Revocação na Base de Dados Completa Derisi.

Analisando a medida de revocação, em todas bases de dados e com todos os limiares, o algoritmo MHC-CNN comparado como o Clus-HMC foi superior na maioria dos casos conforme mostra a Figura 50. Nota-se ainda que com o aumento do número de épocas o percentual da diferença entre os algoritmos também aumenta.

A mesma comparação é feita entre o algoritmo MHC-CNN e o Clus-HSC, conforme mostra a Figura 51, em que o mesmo processo ocorre.

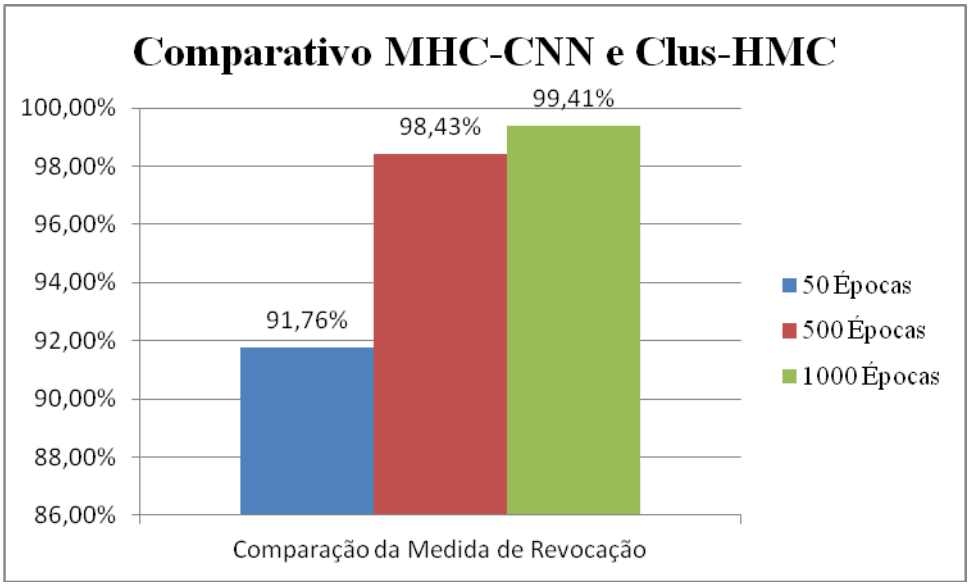


Figura 50 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Base de Dados Completa.

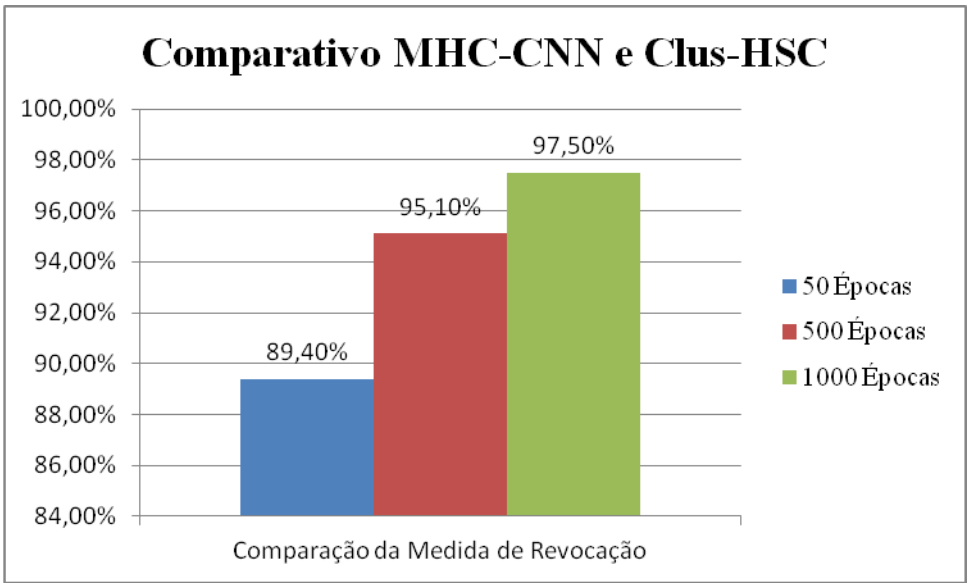


Figura 51 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Base de Dados Completa.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 50 épocas e o Clus-HMC e substituindo os valores na Equação 36 e na Equação 37 tem-se $W^+ = 0$ e $W^- = 126$. Aplicando na Equação 38 tem-se $T = \min(0;126) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada assumindo que há diferença estatística entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 500 épocas e o Clus-HMC. Este resultado indica que a hipótese nula deve ser rejeitada assumindo que a há diferença estatística entre os algoritmos.

Aplicando o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 1000 épocas e o Clus-HMC, obtêm-se $W^+ = 0$ e $W^- = 54$ e $T = \min(0;54) = 0$. Nesse caso também a hipótese nula deve ser rejeitada assumindo que a há diferença significativa entre os algoritmos, pois os valores de W^+ e W^- estão fora do intervalo dado pelos valores do limite inferior e limite superior que são 8 e 47, respectivamente.

6.5.2.2 Base de Dados com Limiar igual a 300

A Tabela 22 apresenta os resultados apresentados na base de dados com limiar igual a 300. Analisando estatisticamente esses resultados através do teste de Friedman $X_F^2 = 32{,}78$ e $F_F = 40{,}9$.

Tabela 22 – Comparativo da Medida AUPRC entre o MHC-CNN e o Clus- HMC e Clus-HSC na Base de Dados com Limiar igual a 300.

	MHC-CNN (em épocas)			Clus	
	50	**500**	**1000**	**HMC**	**HSC**
Cellcycle	0,12	0,07	0,06	0,37	0,25
Church	0,07	0,08	0,07	0,39	0,33
Derisi	0,05	0,07	0,06	0,37	0,31
Eisen	0,02	0,02	0,02	0,74	0,63
Expr	0,07	0,07	0,07	0,42	0,24
Gasch1	0,08	0,06	0,06	0,38	0,23
Gasch2	0,12	0,08	0,07	0,38	0,29
Pheno	0,02	0,03	0,03	0,66	0,38
Seq	0,06	0,05	0,05	0,41	0,24
Spo	0,06	0,05	0,05	0,41	0,31

De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC foi estatisticamente superior ao MHC-CNN, nas três épocas. Já o algoritmo Clus-HSC foi estatisticamente superior ao resultado do MHC-CNN apenas na execução com 1000 épocas.

Assim como nos resultados apresentados anteriormente, também avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados com limiar igual a 300.

Serão apresentados os resultados obtidos com 50, 500 e 1000 épocas. As bases de dados selecionadas também foram a Church e Derisi para seguir o padrão já estabelecido. A Tabela 23 apresenta os resultados obtidos nas duas bases de dados com a execução de 50 épocas. Observa-se que apenas no limiar T(100) o Clus-HMC foi superior ao MHC-CNN. Já

na execução com 500 (Tabela 24) e 1000 épocas (Tabela 25) o MHC-CNN foi superior em todos os limiares considerando a medida de revocação.

Tabela 23 – Comparativo da Medida de Precisão e Revocação com 50 épocas na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**0,90**	0,27	0,54	0,29	0,46	0,01	**0,88**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,87**	0,41	0,43	0,41	0,35	0,01	**0,86**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,86**	0,49	0,36	0,48	0,31	0,01	**0,85**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,85**	0,63	0,25	0,58	0,23	0,01	**0,85**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,84**	0,67	0,22	0,64	0,21	0,01	**0,83**	0,69	0,18	0,63	0,18
T(60)	0,01	**0,81**	0,75	0,16	0,73	0,14	0,01	**0,82**	0,74	0,14	0,71	0,13
T(70)	0,01	**0,75**	0,85	0,11	0,80	0,11	0,01	**0,81**	0,83	0,09	0,79	0,09
T(80)	0,04	**0,57**	0,91	0,09	0,88	0,08	0,01	**0,76**	0,90	0,07	0,92	0,05
T(90)	0,13	**0,27**	0,97	0,07	0,93	0,06	0,05	**0,54**	0,93	0,06	0,95	0,05
T(100)	0,19	0,06	0,98	**0,07**	0,94	0,06	0,10	**0,06**	0,93	**0,06**	0,99	0,04

Tabela 24 – Comparativo da Medida de Precisão e Revocação com 500 épocas na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**0,88**	0,27	0,54	0,29	0,46	0,01	**0,84**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,86**	0,41	0,43	0,41	0,35	0,01	**0,83**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,85**	0,49	0,36	0,48	0,31	0,01	**0,82**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,84**	0,63	0,25	0,58	0,23	0,01	**0,82**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,82**	0,67	0,22	0,64	0,21	0,01	**0,81**	0,69	0,18	0,63	0,18
T(60)	0,02	**0,73**	0,75	0,16	0,73	0,14	0,01	**0,71**	0,74	0,14	0,71	0,13
T(70)	0,04	**0,59**	0,85	0,11	0,80	0,11	0,10	**0,34**	0,83	0,09	0,79	0,09
T(80)	0,05	**0,51**	0,91	0,09	0,88	0,08	0,11	**0,21**	0,90	0,07	0,92	0,05
T(90)	0,07	**0,39**	0,97	0,07	0,93	0,06	0,14	**0,19**	0,93	0,06	0,95	0,05
T(100)	0,23	**0,10**	0,98	0,07	0,94	0,06	0,18	**0,10**	0,93	0,06	0,99	0,04

Tabela 25 – Comparativo da Medida de Precisão e Revocação com 1000 épocas na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**0,88**	0,27	0,54	0,29	0,46	0,01	**0,84**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,86**	0,41	0,43	0,41	0,35	0,01	**0,82**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,84**	0,49	0,36	0,48	0,31	0,01	**0,82**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,84**	0,63	0,25	0,58	0,23	0,01	**0,82**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,82**	0,67	0,22	0,64	0,21	0,01	**0,81**	0,69	0,18	0,63	0,18
T(60)	0,02	**0,74**	0,75	0,16	0,73	0,14	0,02	**0,66**	0,74	0,14	0,71	0,13
T(70)	0,04	**0,62**	0,85	0,11	0,80	0,11	0,12	**0,30**	0,83	0,09	0,79	0,09
T(80)	0,04	**0,55**	0,91	0,09	0,88	0,08	0,11	**0,22**	0,90	0,07	0,92	0,05
T(90)	0,06	**0,42**	0,97	0,07	0,93	0,06	0,11	**0,20**	0,93	0,06	0,95	0,05
T(100)	0,23	**0,08**	0,98	0,07	0,94	0,06	0,18	**0,10**	0,93	0,06	0,99	0,04

Para melhor visualização dos resultados mostrados nas três tabelas anteriores a Figura 52 apresenta os resultados da base de dados Church e a Figura 53 os resultados obtidos da base de dados Derisi.

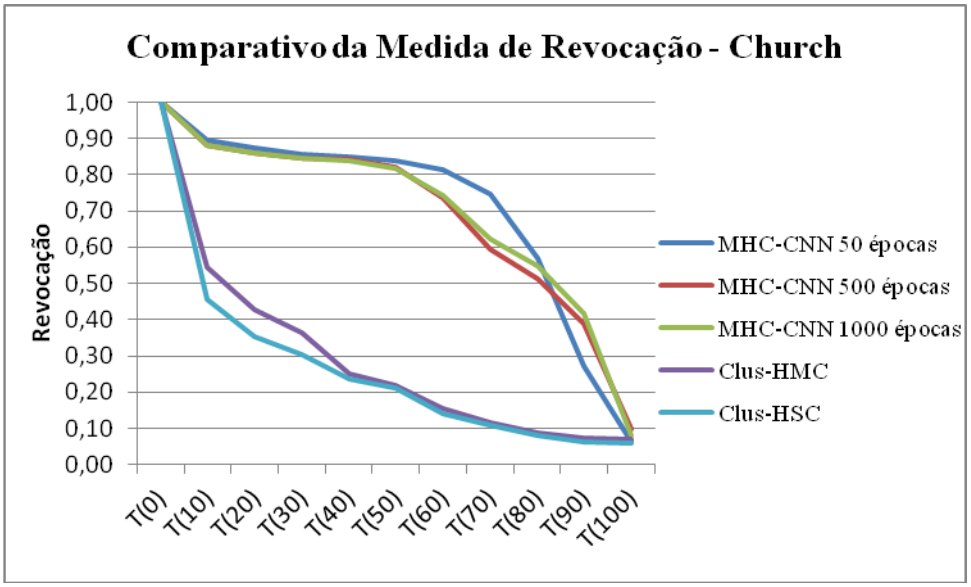


Figura 52 – Comparativo Geral da Medida de Revocação na Base de Dados Church com Limiar igual a 300.

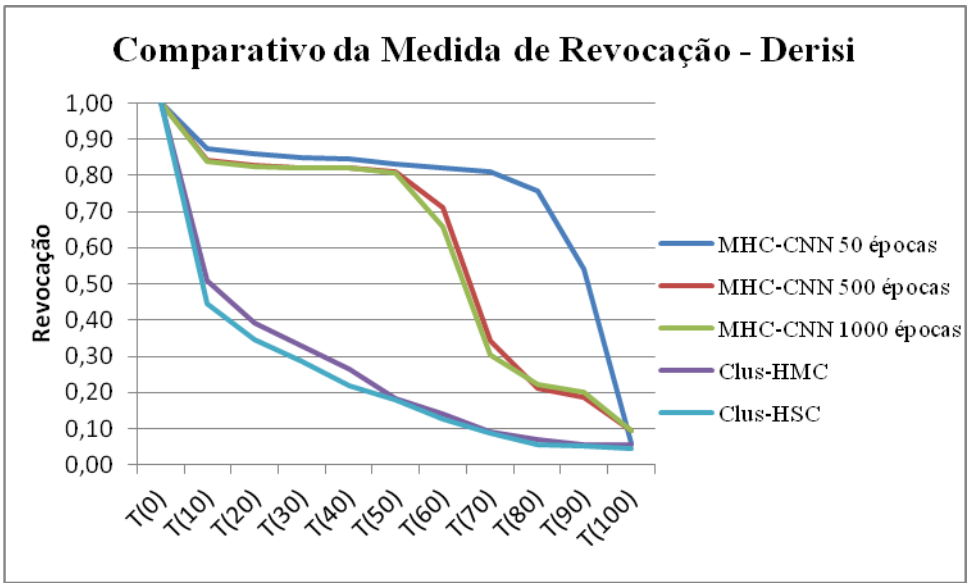


Figura 53 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi com Limiar igual a 300.

Analisando a medida de revocação, em todas bases de dados e com todos os limiares, o algoritmo MHC-CNN comparado com o Clus-HMC foi superior em mais de 94% dos casos conforme mostra a Figura 54.

A mesma comparação é feita entre o algoritmo MHC-CNN e o Clus-HSC, conforme mostra a Figura 55. Nota-se, que neste caso, que como o aumento do número de épocas a quantidade avaliada da medida de revocação também aumenta.

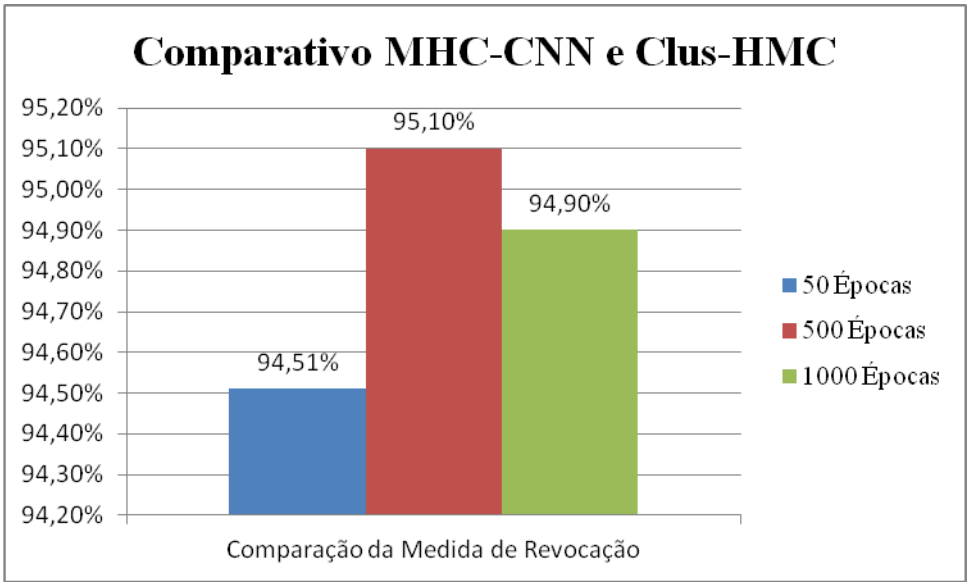


Figura 54 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Base com Limiar igual a 300.

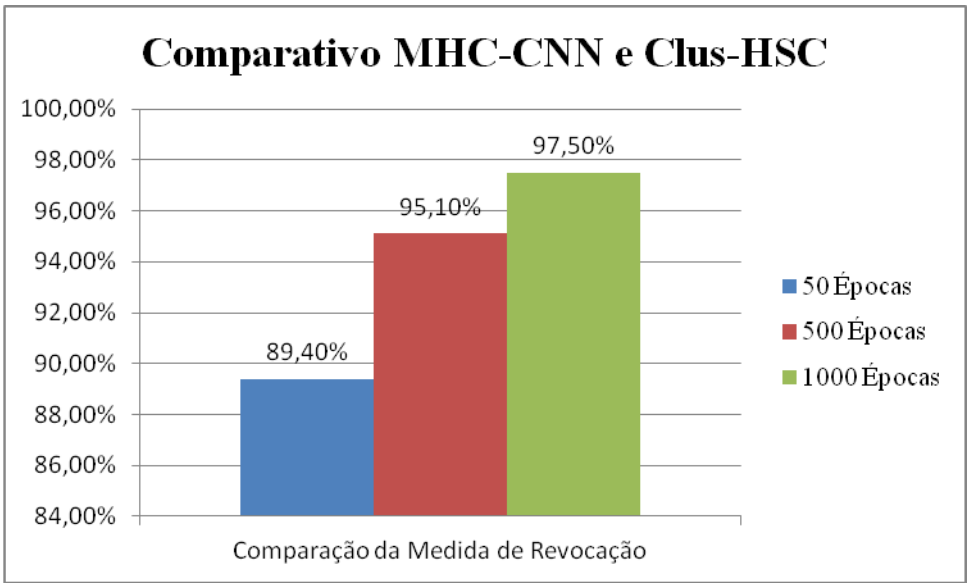


Figura 55 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Base de Dados com Limiar igual a 300.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 50 épocas e o Clus HMC e substituindo os valores nas Equação 36 e Equação 37Equação 38 têm-se $W^{+}=0$ e $W^{-}=55$. Aplicando na Equação 37 tem-se $T=\min(0;55)=0$.

Considerando que o valor de $n(10<25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha=0{,}05$ a hipótese nula deve ser rejeitada, pois está fora do intervalo dado pelos valores do limite inferior e limite superior que são 8 e 47, respectivamente, assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 500 épocas e 1000 épocas.

6.5.2.3 Base de Dados Separadas por Ontologias

Assim como na comparação entre os algoritmos MHC-CNN e MHC-ES serão apresentados os resultados obtidos da ontologia Componente Celular. Os resultados das duas outras ontologias: Função Molecular e Processo Biológico serão mostrados no Apêndice B.

A Tabela 26 mostra os resultados dos algoritmos utilizando a medida AUPRC com 50, 500 e 1000 épocas. Analisando estatisticamente esses resultados e recalculando os valores para as equações utilizadas no teste de Friedman, como os dados apresentados têm-se $X_F^2=32{,}0$ e $F_F=36{,}1$.

Tabela 26 – Comparativo da Medida AUPRC entre o MHC-CNN e o Clus-HMC e Clus-HSC na Ontologia Componente Celular.

	MHC-CNN (em épocas)			Clus	
	50	**500**	**1000**	**HMC**	**HSC**
Cellcycle	0,41	0,30	0,27	0,62	0,53
Church	0,27	0,28	0,27	0,63	0,57
Derisi	0,27	0,27	0,30	0,62	0,55
Eisen	0,28	0,28	0,29	0,64	0,54
Expr	0,26	0,27	0,26	0,68	0,48
Gasch1	0,12	0,12	0,12	0,76	0,69
Gasch2	0,39	0,29	0,29	0,63	0,55
Pheno	0,23	0,23	0,26	0,61	0,58
Seq	0,26	0,26	0,26	0,66	0,46
Spo	0,26	0,26	0,26	0,67	0,56

De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC e Clus-HSC foi estatisticamente superior ao MHC-CNN nas três épocas (50, 500 e 1000).

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados do domínio Componente Celular.

Serão apresentados os resultados obtidos com 50, 500 e 1000 épocas. As bases de dados selecionadas também foram a Church e Derisi para seguir o padrão já estabelecido. A Tabela 27 apresenta os resultados obtidos nas duas bases de dados com a execução de 50 épocas. Observa-se que em todos os limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. O mesmo ocorre na execução com 500 épocas conforme mostra a Tabela 28 e na execução com 1000 épocas como mostra a Tabela 29.

Tabela 27 – Comparativo da Medida de Precisão e Revocação com 50 épocas na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,03	**0,85**	0,43	0,75	0,47	0,66	0,04	**0,84**	0,43	0,75	0,46	0,65
T(20)	0,04	**0,83**	0,51	0,68	0,57	0,57	0,05	**0,82**	0,51	0,68	0,54	0,56
T(30)	0,05	**0,82**	0,60	0,60	0,63	0,48	0,05	**0,81**	0,58	0,62	0,60	0,47
T(40)	0,05	**0,82**	0,63	0,56	0,71	0,41	0,06	**0,80**	0,63	0,53	0,66	0,42
T(50)	0,06	**0,80**	0,82	0,35	0,81	0,34	0,07	**0,79**	0,73	0,41	0,72	0,37
T(60)	0,09	**0,76**	0,89	0,31	0,85	0,32	0,09	**0,78**	0,91	0,28	0,85	0,29
T(70)	0,14	**0,72**	0,92	0,30	0,88	0,30	0,11	**0,77**	0,93	0,27	0,88	0,27
T(80)	0,28	**0,58**	0,93	0,28	0,89	0,28	0,19	**0,70**	0,93	0,27	0,89	0,26
T(90)	0,33	**0,55**	0,95	0,22	0,92	0,23	0,40	**0,47**	0,97	0,19	0,92	0,23
T(100)	0,41	**0,46**	0,99	0,11	0,96	0,11	0,40	**0,46**	1,00	0,10	0,96	0,17

Tabela 28 – Comparativo da Medida de Precisão e Revocação com 500 épocas na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,03	**0,84**	0,43	0,75	0,47	0,66	0,06	**0,80**	0,43	0,75	0,46	0,65
T(20)	0,04	**0,82**	0,51	0,68	0,57	0,57	0,09	**0,78**	0,51	0,68	0,54	0,56
T(30)	0,05	**0,81**	0,60	0,60	0,63	0,48	0,10	**0,77**	0,58	0,62	0,60	0,47
T(40)	0,05	**0,81**	0,63	0,56	0,71	0,41	0,10	**0,77**	0,63	0,53	0,66	0,42
T(50)	0,06	**0,81**	0,82	0,35	0,81	0,34	0,11	**0,77**	0,73	0,41	0,72	0,37
T(60)	0,07	**0,79**	0,89	0,31	0,85	0,32	0,15	**0,73**	0,91	0,28	0,85	0,29
T(70)	0,09	**0,68**	0,92	0,30	0,88	0,30	0,34	**0,52**	0,93	0,27	0,88	0,27
T(80)	0,12	**0,66**	0,93	0,28	0,89	0,28	0,36	**0,52**	0,93	0,27	0,89	0,26
T(90)	0,19	**0,59**	0,95	0,22	0,92	0,23	0,39	**0,48**	0,97	0,19	0,92	0,23
T(100)	0,46	**0,45**	0,99	0,11	0,96	0,11	0,40	**0,46**	1,00	0,10	0,96	0,17

Com objetivo de comparar os resultados obtidos e mostrados nas três tabelas anteriores a Figura 56 apresenta os resultados da base de dados Church e a Figura 57 os resultados obtidos da base de dados Derisi.

Tabela 29 – Comparativo da Medida de Precisão e Revocação com 1000 épocas na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,03	**0,83**	0,43	0,75	0,47	0,66	0,07	**0,79**	0,43	0,75	0,46	0,65
T(20)	0,05	**0,81**	0,51	0,68	0,57	0,57	0,09	**0,77**	0,51	0,68	0,54	0,56
T(30)	0,05	**0,81**	0,60	0,60	0,63	0,48	0,10	**0,77**	0,58	0,62	0,60	0,47
T(40)	0,06	**0,80**	0,63	0,56	0,71	0,41	0,11	**0,77**	0,63	0,53	0,66	0,42
T(50)	0,06	**0,80**	0,82	0,35	0,81	0,34	0,11	**0,76**	0,73	0,41	0,72	0,37
T(60)	0,07	**0,78**	0,89	0,31	0,85	0,32	0,11	**0,75**	0,91	0,28	0,85	0,29
T(70)	0,09	**0,68**	0,92	0,30	0,88	0,30	0,21	**0,64**	0,93	0,27	0,88	0,27
T(80)	0,11	**0,66**	0,93	0,28	0,89	0,28	0,56	**0,33**	0,93	0,27	0,89	0,26
T(90)	0,16	**0,62**	0,95	0,22	0,92	0,23	0,56	**0,33**	0,97	0,19	0,92	0,23
T(100)	0,44	**0,43**	0,99	0,11	0,96	0,11	0,56	**0,33**	1,00	0,10	0,96	0,17

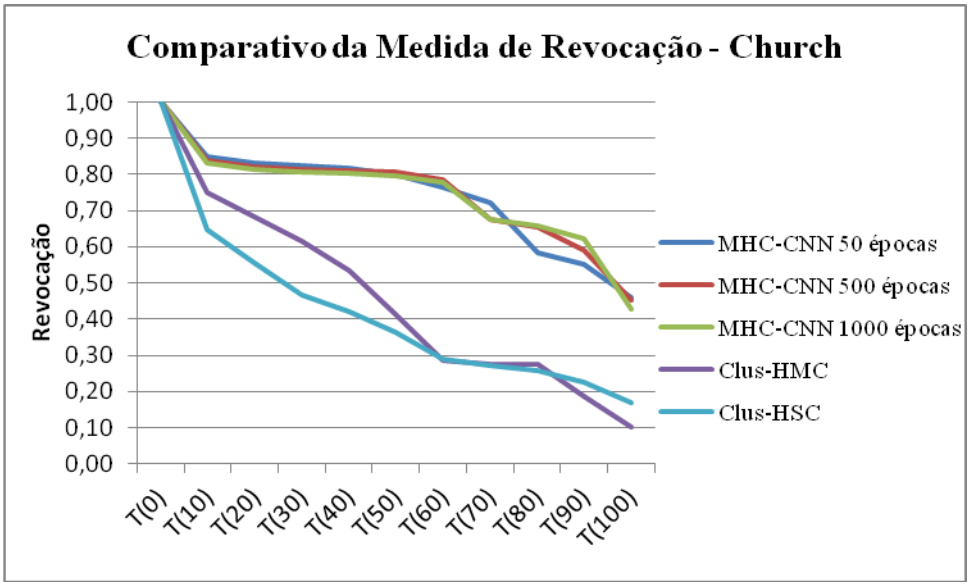


Figura 56 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Componente Celular.

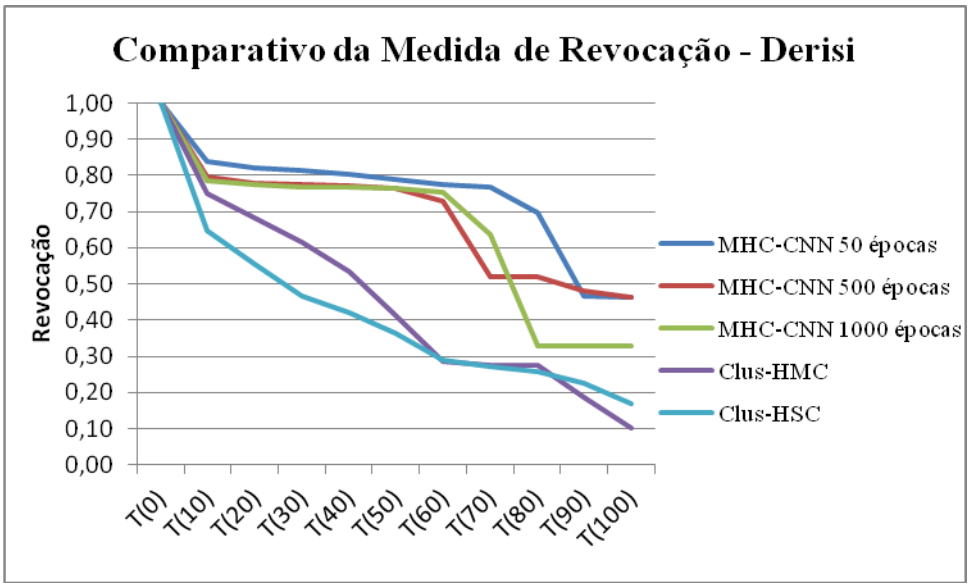


Figura 57 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Componente Celular.

Analisando a medida de revocação, em todas as bases de dados da ontologia Componente Celular e com todos os limiares, o algoritmo MHC-CNN comparado como o Clus-HMC foi superior na maioria dos casos conforme mostra a Figura 58. Nota-se, neste caso que com o aumento do número de épocas o percentual da diferença entre os algoritmos diminuiu.

A mesma comparação é feita entre o algoritmo MHC-CNN e o Clus-HSC, conforme mostra a Figura 59, porém, neste caso com o aumento do número de épocas o percentual da diferença entre os algoritmos também aumenta.

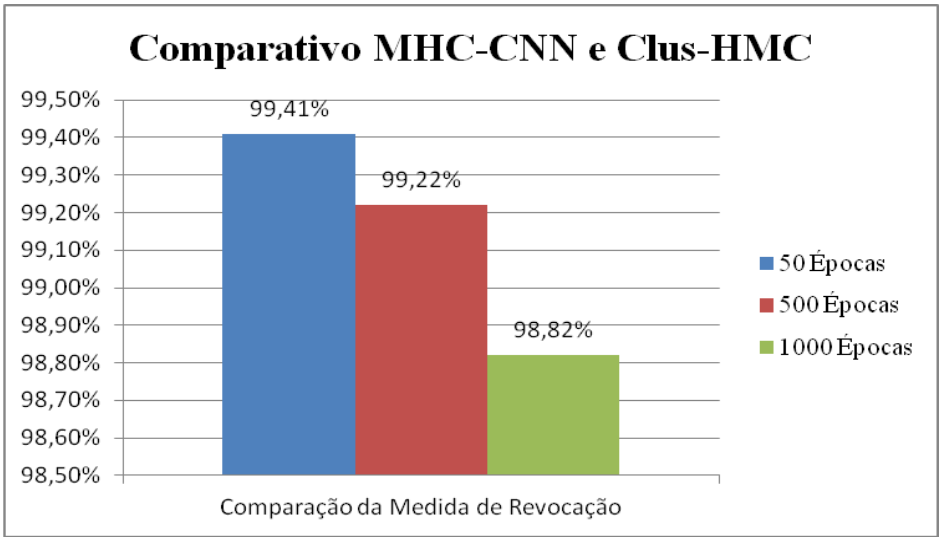


Figura 58 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Ontologia Componente Celular.

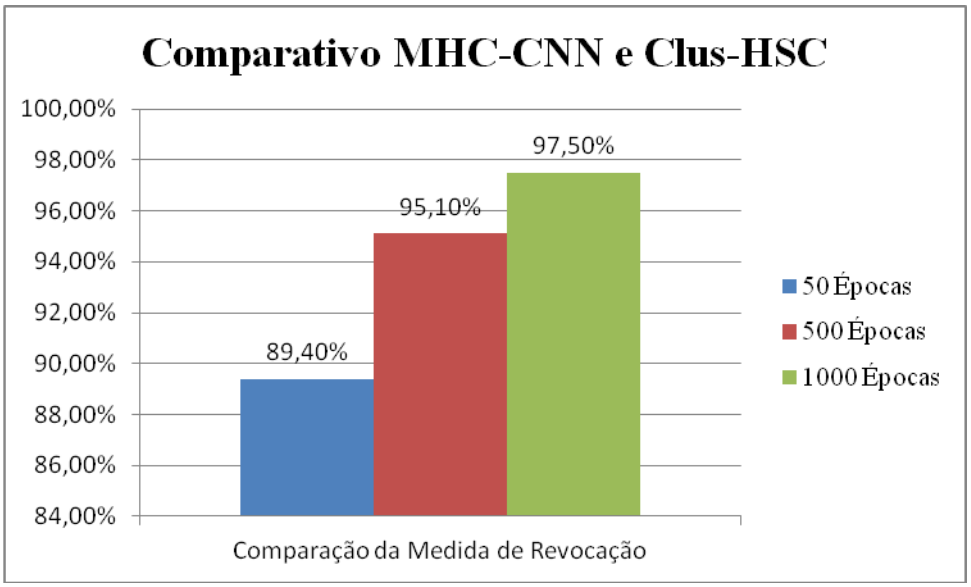


Figura 59 – Comparativo da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Ontologia Componente Celular.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 50 épocas e o Clus-HMC e substituindo os valores nas Equação 36 e Equação 37 têm-se $W^{+}=0$ e $W^{-}=55$. Aplicando na Equação 38 tem-se $T=\min(0;55)=0$.

Considerando que o valor de $n(10<25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha=0{,}05$ a hipótese nula deve ser rejeitada, pois está fora do intervalo dado pelos valores do limite inferior e limite superior que são 8 e 47, respectivamente, assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 500 épocas e 1000 épocas.

6.5.3 Comparativo entre MHC-ES e o Clus

Da mesma maneira que foi avaliada a comparação entre os algoritmos MHC-CNN e o Clus-HMC e Clus-HSC será feita a comparação entre os algoritmos MHC-ES e o Clus-HMC e Clus-HSC. Por questões de espaço são mostrados os resultados obtidos com 20, 60 e 100 gerações.

6.5.3.1 Base de Dados Completa

A Tabela 30 apresenta os resultados apresentados na base de dados completa. Analisando estatisticamente esses resultados e substituindo os valores para as equações utilizadas no teste de Friedman têm-se $X_F^2 = 32{,}26$ e $F_F = 37{,}5$.

Tabela 30 – Comparativo da Medida AUPRC entre o MHC-ES e o Clus-HMC e Clus-HSC na Base de Dados Completa.

	MHC-ES (em gerações)			Clus	
	20	**60**	**100**	**HMC**	**HSC**
Cellcycle	0,06	0,07	0,06	0,44	0,32
Church	0,03	0,03	0,03	0,45	0,38
Derisi	0,06	0,06	0,07	0,44	0,35
Eisen	0,06	0,05	0,04	0,46	0,35
Expr	0,03	0,03	0,03	0,47	0,28
Gasch1	0,04	0,03	0,04	0,46	0,31
Gasch2	0,03	0,03	0,03	0,44	0,34
Pheno	0,08	0,08	0,12	0,42	0,39
Seq	0,06	0,06	0,12	0,47	0,28
Spo	0,06	0,08	0,07	0,47	0,36

De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC foi estatisticamente superior ao MHC-CNN nas três épocas selecionadas (50, 500 e 1000). Já o algoritmo Clus-HSC foi estatisticamente superior ao MHC-CNN com 50 e 1000 épocas.

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Para fins de demonstração, selecionaram-se duas bases de dados para apresentar os resultados obtidos com 20, 60 e 100 gerações. As bases de dados selecionadas foram as mesmas selecionadas para a comparação dos algoritmos MHC-CNN e Clus: Church e Derisi.

A Tabela 31 apresenta os resultados obtidos nas duas bases de dados com a execução de 20 gerações. Observa-se que nos limiares de T(0) a T(60) os resultados da medida de

revocação do algoritmo MHC-ES foi superior, identificados em negrito, comparado com os dois outros algoritmos na base de dados Church. No que se refere à base de dados Derisi nos limiares de T(0) a T(70) os resultados da medida de revocação do algoritmo MHC-ES foi superior.

Tabela 31 – Comparativo da Medida de Precisão e Revocação com 20 gerações na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,32	0,55	0,34	0,46	0,02	**0,86**	0,32	0,54	0,30	0,44
T(20)	0,01	**0,99**	0,49	0,44	0,49	0,38	0,02	**0,81**	0,49	0,43	0,43	0,37
T(30)	0,01	**0,94**	0,59	0,38	0,60	0,31	0,02	**0,79**	0,61	0,36	0,53	0,30
T(40)	0,01	**0,82**	0,67	0,33	0,67	0,27	0,04	**0,71**	0,67	0,31	0,62	0,27
T(50)	0,02	**0,61**	0,80	0,25	0,75	0,23	0,06	**0,62**	0,79	0,24	0,70	0,23
T(60)	0,02	**0,32**	0,89	0,20	0,83	0,20	0,07	**0,46**	0,86	0,20	0,78	0,20
T(70)	0,06	0,05	0,92	**0,19**	0,86	**0,19**	0,07	**0,19**	0,92	0,17	0,83	0,18
T(80)	0,08	0,04	0,94	**0,17**	0,89	0,16	0,04	0,06	0,93	0,16	0,87	**0,17**
T(90)	0,13	0,04	0,98	0,13	0,93	**0,14**	0,07	0,05	0,97	0,12	0,92	**0,14**
T(100)	0,20	0,03	0,99	**0,10**	0,95	**0,10**	0,24	0,03	1,00	**0,09**	0,96	**0,09**

Já na execução do algoritmo MHC-ES com 60 gerações, conforme mostra a

Tabela 32 os resultados nas duas bases de dados, foi superior aos demais algoritmos nos limiares T(0) a T(60).

Tabela 32 – Comparativo da Medida de Precisão e Revocação com 60 gerações na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,32	0,55	0,34	0,46	0,02	**0,84**	0,32	0,54	0,30	0,44
T(20)	0,01	**0,99**	0,49	0,44	0,49	0,38	0,02	**0,80**	0,49	0,43	0,43	0,37
T(30)	0,01	**0,94**	0,59	0,38	0,60	0,31	0,03	**0,75**	0,61	0,36	0,53	0,30
T(40)	0,01	**0,82**	0,67	0,33	0,67	0,27	0,04	**0,58**	0,67	0,31	0,62	0,27
T(50)	0,02	**0,59**	0,80	0,25	0,75	0,23	0,07	**0,41**	0,79	0,24	0,70	0,23
T(60)	0,02	**0,30**	0,89	0,20	0,83	0,20	0,08	**0,23**	0,86	0,20	0,78	0,20
T(70)	0,06	0,06	0,92	**0,19**	0,86	**0,19**	0,09	0,06	0,92	0,17	0,83	**0,18**
T(80)	0,09	0,04	0,94	**0,17**	0,89	0,16	0,13	0,04	0,93	0,16	0,87	**0,17**
T(90)	0,13	0,04	0,98	0,13	0,93	**0,14**	0,17	0,04	0,97	0,12	0,92	**0,14**
T(100)	0,20	0,03	0,99	**0,10**	0,95	**0,10**	0,20	0,03	1,00	**0,09**	0,96	**0,09**

O mesmo ocorre na execução com a execução do algoritmo com 100 gerações conforme mostra a Tabela 33.

Tabela 33 – Comparativo da Medida de Precisão e Revocação com 100 gerações na Base de Dados Completa.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,32	0,55	0,34	0,46	0,02	**0,85**	0,32	0,54	0,30	0,44
T(20)	0,01	**0,99**	0,49	0,44	0,49	0,38	0,02	**0,82**	0,49	0,43	0,43	0,37
T(30)	0,01	**0,96**	0,59	0,38	0,60	0,31	0,03	**0,79**	0,61	0,36	0,53	0,30
T(40)	0,01	**0,86**	0,67	0,33	0,67	0,27	0,04	**0,67**	0,67	0,31	0,62	0,27
T(50)	0,01	**0,62**	0,80	0,25	0,75	0,23	0,07	**0,48**	0,79	0,24	0,70	0,23
T(60)	0,02	**0,30**	0,89	0,20	0,83	0,20	0,08	**0,23**	0,86	0,20	0,78	0,20
T(70)	0,07	0,07	0,92	**0,19**	0,86	**0,19**	0,16	0,11	0,92	0,17	0,83	**0,18**
T(80)	0,09	0,04	0,94	**0,17**	0,89	0,16	0,10	0,04	0,93	0,16	0,87	**0,17**
T(90)	0,13	0,04	0,98	0,13	0,93	**0,14**	0,17	0,04	0,97	0,12	0,92	**0,14**
T(100)	0,20	0,03	0,99	**0,10**	0,95	**0,10**	0,20	0,03	1,00	**0,09**	0,96	**0,09**

Com objetivo de comparar os resultados obtidos e mostrados nas três tabelas anteriores a Figura 60 apresenta os resultados da base de dados Church e a Figura 61 os resultados obtidos da base de dados Derisi.

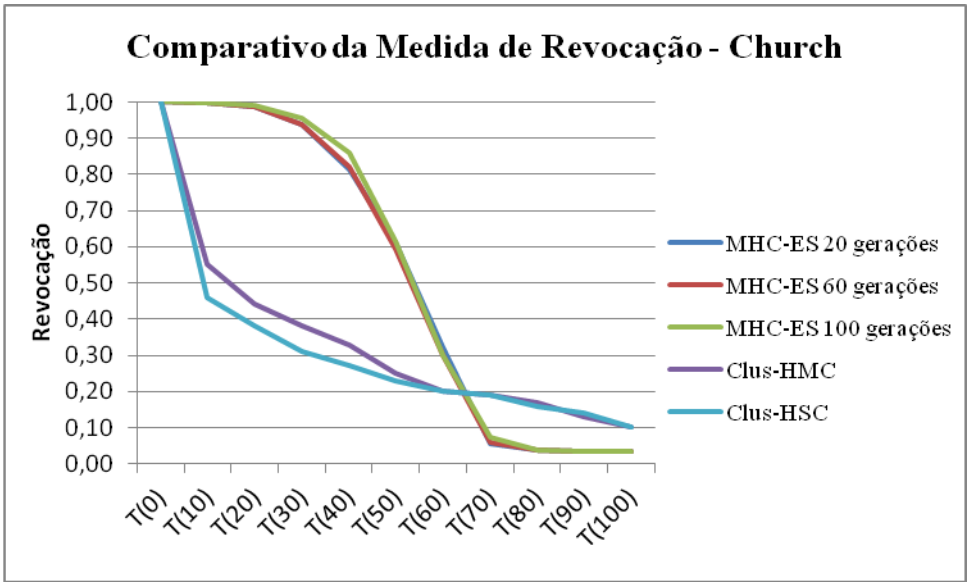


Figura 60 – Comparativo Geral da Medida de Revocação na Base de Dados Church Completa.

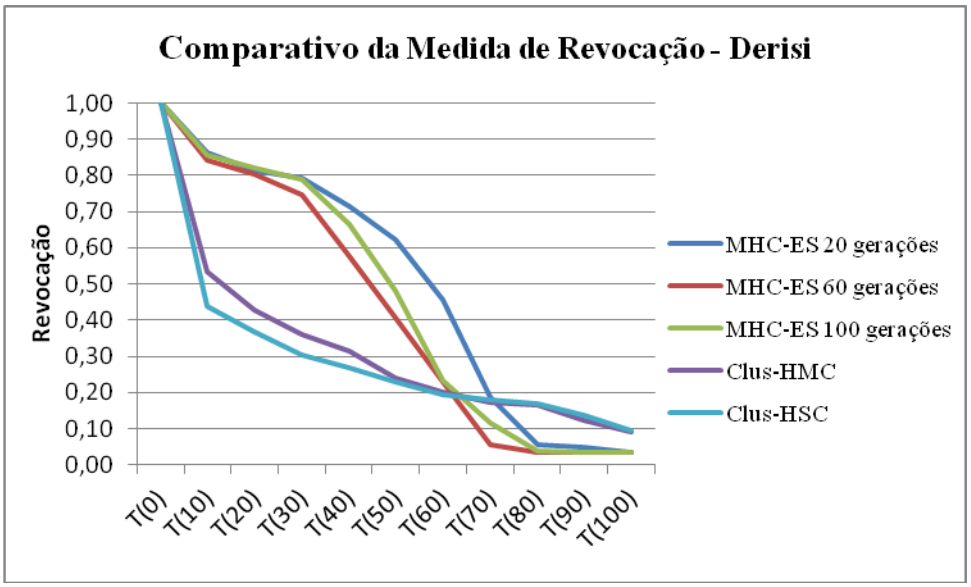


Figura 61 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi Completa.

Analisando a medida de revocação, em todas as bases de dados e com todos os limiares, o algoritmo MHC-ES comparado como o Clus-HMC foi inferior na maioria dos casos conforme mostra a Figura 62.

A mesma comparação é feita entre o algoritmo MHC-ES e o Clus-HSC, conforme mostra a Figura 63, em que o mesmo processo ocorre.

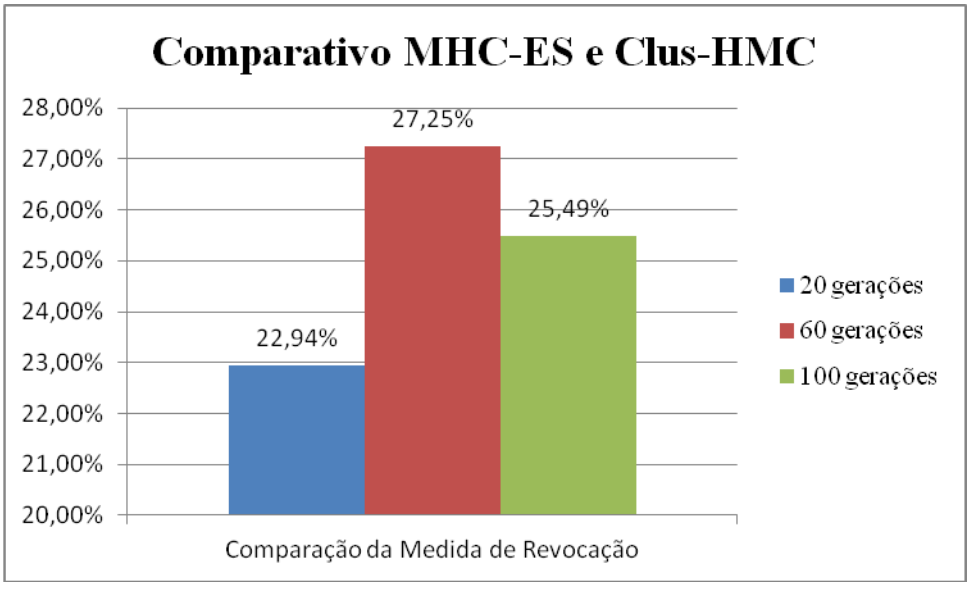


Figura 62 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HMC.

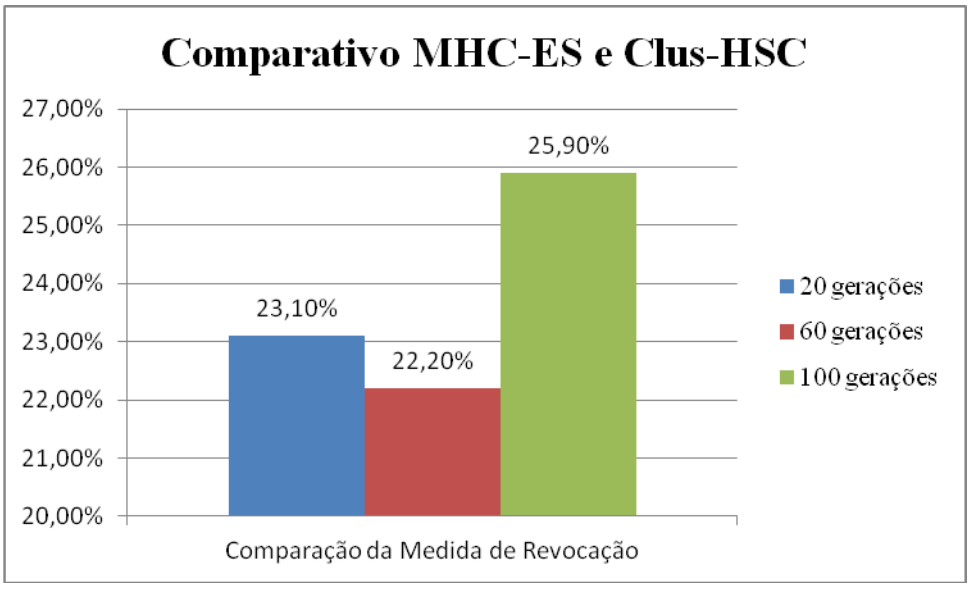


Figura 63 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HSC na Base de Dados Completa.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 20 gerações e o Clus-HMC, obtêm $W^+ = 0$ e $W^- = 56$. Logo se tem $T = \min(0;56) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada, pois está fora do intervalo dado pelos valores do limite inferior e limite superior de 8 e 47, respectivamente.

O resultado obtido pelo algoritmo MHC-ES, com 60 e 100 gerações, quando aplicado o teste de Wilcoxon tiveram o mesmo resultado, sendo $W^+ = 0$ e $W^- = 55$. Aplicando na Equação 39 tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula também deve ser rejeitada, pois está fora do intervalo dado pelos valores do limite inferior e limite superior de 8 e 47, respectivamente.

6.5.3.2 Base de Dados com Limiar igual a 300

A Tabela 34 apresenta os resultados apresentados com a base de dados com limiar igual a 300. Analisando estatisticamente esses resultados e recalculando os valores para as equações utilizadas no teste de Friedman têm-se $X_F^2 = 32{,}18$ e $F_F = 37{,}0$.

Tabela 34 – Comparativo da Medida AUPRC entre o MHC-ES e o Clus-HMC e Clus-HSC na Base de Dados com Limiar igual a 300.

	MHC-ES (em gerações)			Clus	
	20	**60**	**100**	**HMC**	**HSC**
Cellcycle	0,06	0,04	0,05	0,37	0,25
Church	0,02	0,02	0,02	0,39	0,33
Derisi	0,04	0,05	0,05	0,37	0,31
Eisen	0,01	0,01	0,01	0,74	0,63
Expr	0,03	0,03	0,03	0,42	0,24
Gasch1	0,05	0,03	0,02	0,38	0,23
Gasch2	0,03	0,02	0,03	0,38	0,29
Pheno	0,01	0,02	0,01	0,66	0,38
Seq	0,06	0,06	0,12	0,41	0,24
Spo	0,06	0,05	0,05	0,41	0,31

De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Através do valor obtido pela $CD = 1{,}93$, considerando o nível de significância de 95%, observa-se que o algoritmo Clus-HMC foi estatisticamente superior ao MHC-CNN nas três épocas selecionadas (50, 500 e 1000). Já o algoritmo Clus-HSC foi estatisticamente superior ao MHC-CNN com 500 e 1000 épocas.

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Também foram selecionadas duas bases de dados para apresentar os resultados obtidos com 20, 60 e 100 gerações. As bases de dados selecionadas foram as mesmas selecionadas para a comparação dos algoritmos MHC-ES e Clus: Church e Derisi.

A Tabela 35 apresenta os resultados obtidos nas duas bases de dados com a execução de 20 gerações. Observa-se que, na base de dados Church, nos limiares de T(0) a T(60) os resultados da medida de revocação do algoritmo MHC-ES foi superior, identificados em negrito, comparado com os dois outros algoritmos. No que se refere a base de dados Derisi nos limiares de T(0) a T(90) os resultados da medida de revocação do algoritmo MHC-ES foi superior.

Tabela 35 – Comparativo da Medida de Precisão e Revocação com 20 gerações na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,27	0,54	0,29	0,46	0,01	**0,87**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,99**	0,41	0,43	0,41	0,35	0,01	**0,82**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,94**	0,49	0,36	0,48	0,31	0,02	**0,80**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,84**	0,63	0,25	0,58	0,23	0,02	**0,70**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,63**	0,67	0,22	0,64	0,21	0,04	**0,58**	0,69	0,18	0,63	0,18
T(60)	0,01	**0,31**	0,75	0,16	0,73	0,14	0,05	**0,39**	0,74	0,14	0,71	0,13
T(70)	0,03	0,05	0,85	**0,11**	0,80	**0,11**	0,08	**0,26**	0,83	**0,09**	0,79	**0,09**
T(80)	0,04	0,03	0,91	**0,09**	0,88	0,08	0,05	**0,07**	0,90	**0,07**	0,92	0,05
T(90)	0,06	0,03	0,97	**0,07**	0,93	0,06	0,07	**0,06**	0,93	**0,06**	0,95	0,05
T(100)	0,08	0,03	0,98	**0,07**	0,94	0,06	0,10	0,03	0,93	**0,06**	0,99	0,04

Já na execução do algoritmo MHC-ES com 60 gerações, conforme mostra a Tabela 36 os resultados da base de dados Church foi igual o que acorreu na execução com 20 gerações, enquanto que na base de dados Derisi a medida de revocação teve resultado superior aos demais algoritmos até o limiar T(50) como mostra a Tabela 36. O mesmo resultado foi obtido com a execução do algoritmo MHC-ES com 100 gerações. Isto pode ser visualizado na Tabela 37.

Tabela 36 – Comparativo da Medida de Precisão e Revocação com 60 gerações na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,27	0,54	0,29	0,46	0,01	**0,85**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,99**	0,41	0,43	0,41	0,35	0,02	**0,82**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,94**	0,49	0,36	0,48	0,31	0,02	**0,75**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,85**	0,63	0,25	0,58	0,23	0,04	**0,61**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,59**	0,67	0,22	0,64	0,21	0,06	**0,27**	0,69	0,18	0,63	0,18
T(60)	0,01	**0,32**	0,75	0,16	0,73	0,14	0,14	0,07	0,74	0,14	0,71	0,13
T(70)	0,03	0,04	0,85	0,11	0,80	0,11	0,12	0,03	0,83	0,09	0,79	0,09
T(80)	0,05	0,04	0,91	0,09	0,88	0,08	0,12	0,03	0,90	0,07	0,92	0,05
T(90)	0,10	0,03	0,97	0,07	0,93	0,06	0,12	0,03	0,93	0,06	0,95	0,05
T(100)	0,13	0,03	0,98	0,07	0,94	0,06	0,12	0,03	0,93	0,06	0,99	0,04

Tabela 37 – Comparativo da Medida de Precisão e Revocação com 100 gerações na Base de Dados com Limiar igual a 300.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,27	0,54	0,29	0,46	0,01	**0,85**	0,30	0,51	0,26	0,45
T(20)	0,01	**0,99**	0,41	0,43	0,41	0,35	0,02	**0,81**	0,45	0,39	0,38	0,35
T(30)	0,01	**0,95**	0,49	0,36	0,48	0,31	0,02	**0,71**	0,53	0,33	0,48	0,29
T(40)	0,01	**0,85**	0,63	0,25	0,58	0,23	0,05	**0,49**	0,59	0,26	0,56	0,22
T(50)	0,01	**0,64**	0,67	0,22	0,64	0,21	0,08	**0,23**	0,69	0,18	0,63	0,18
T(60)	0,01	**0,27**	0,75	0,16	0,73	0,14	0,07	0,05	0,74	**0,14**	0,71	0,13
T(70)	0,04	0,05	0,85	**0,11**	0,80	**0,11**	0,12	0,03	0,83	**0,09**	0,79	**0,09**
T(80)	0,05	0,04	0,91	**0,09**	0,88	0,08	0,12	0,03	0,90	**0,07**	0,92	0,05
T(90)	0,09	0,03	0,97	**0,07**	0,93	0,06	0,12	0,03	0,93	**0,06**	0,95	0,05
T(100)	0,13	0,03	0,98	**0,07**	0,94	0,06	0,12	0,03	0,93	**0,06**	0,99	0,04

Analisando e comparando os resultados obtidos e mostrados na Tabela 35, na Tabela 36 e na Tabela 37 a Figura 64 apresenta os resultados da base de dados Church e a Figura 65 os resultados obtidos da base de dados Derisi.

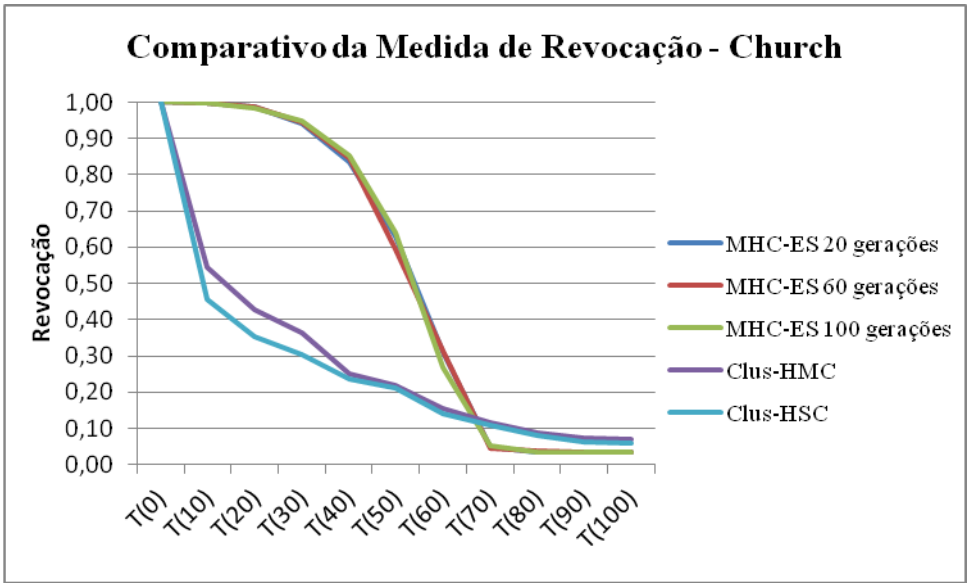


Figura 64 – Comparativo Geral da Medida de Revocação na Base de Dados Church com Limiar igual a 300.

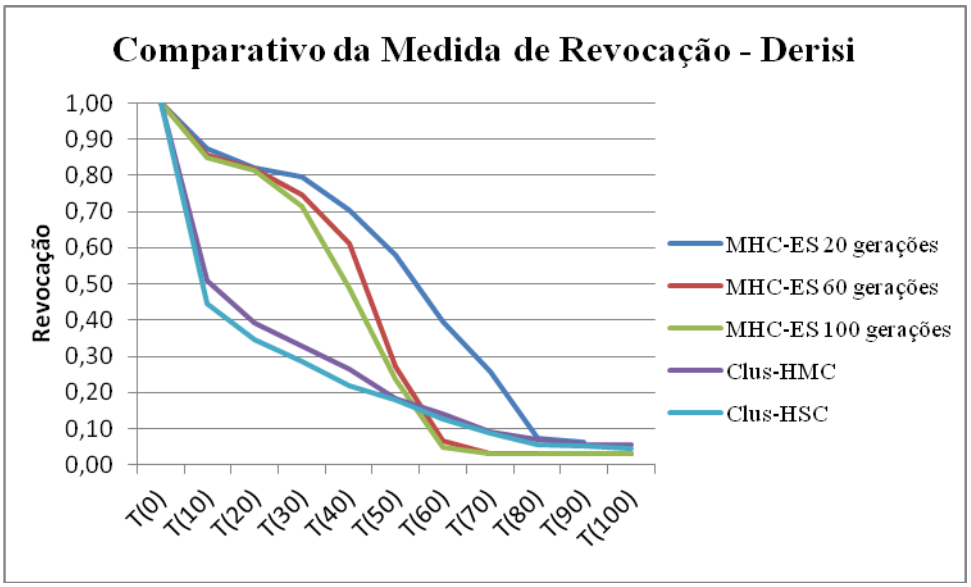


Figura 65 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi com Limiar igual a 300.

Analisando a medida de revocação, em todas as bases de dados e com todos os limiares, o algoritmo MHC-ES comparado como o Clus-HMC foi inferior na maioria dos casos conforme mostra a Figura 66. A mesma comparação é feita entre o algoritmo MHC-ES e o Clus-HSC, conforme mostra a Figura 67, em que o mesmo processo ocorre.

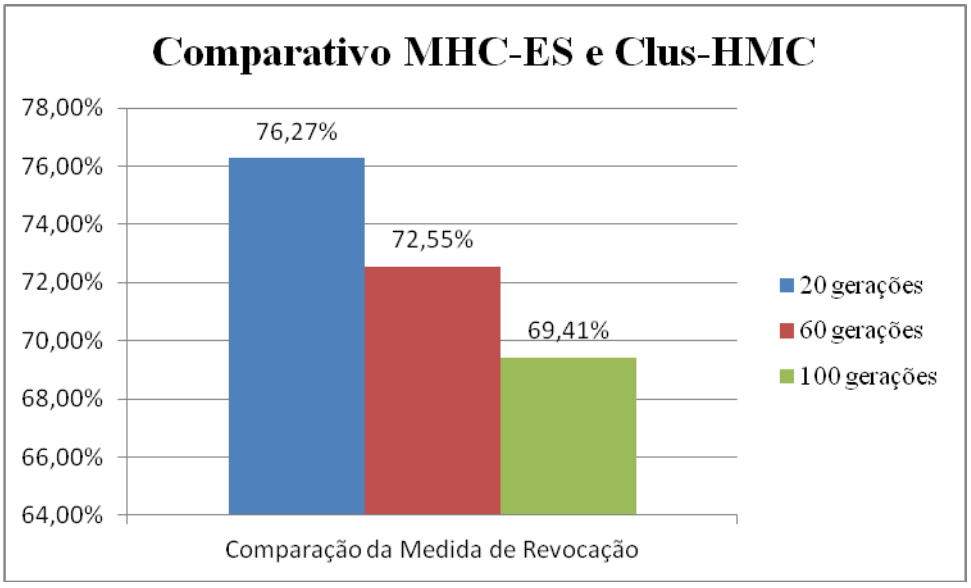


Figura 66 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HMC na Base de Dados com Limiar igual a 300.

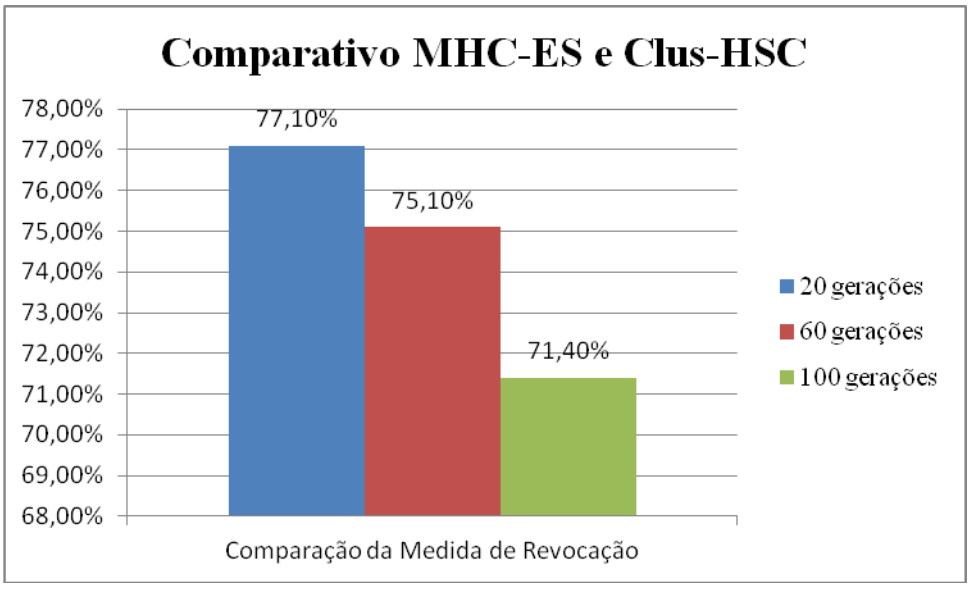


Figura 67 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HSC na Base de Dados com Limiar igual a 300.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 20 gerações e o Clus-HMC, obtêm $W^+ = 0$ e $W^- = 55$. Logo, tem-se $T = \min(0;55) = 0$. Neste caso, a hipótese nula deve ser rejeitada, pois está fora do intervalo dado pelos valores do limite inferior e limite superior que são 8 e 47, respectivamente.

O mesmo resultado foi obtido pelo algoritmo MHC-ES, com 60 e 100 gerações, quando aplicado o teste de Wilcoxon.

6.5.3.3 Base de Dados Separadas por Ontologias

Assim como nas comparações anteriores serão apresentados os resultados obtidos da ontologia Componente Celular. Os resultados das duas outras ontologias: Função Molecular e Processo Biológico serão mostrados no Apêndice B.

A Tabela 38 mostra os resultados dos algoritmos utilizando a medida AUPRC com 20, 60 e 100 gerações.

Calculando os valores para as equações utilizadas no teste de Friedman, como os dados apresentados na Tabela 38, têm-se $X_F^2 = 32{,}98$ e $F_F = 42{,}3$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Tabela 38 – Comparativo da Medida AUPRC entre o MHC-ES e o Clus-HMC e Clus-HSC na Ontologia Componente Celular.

	MHC-ES (em gerações)			Clus	
	20	**60**	**100**	**HMC**	**HSC**
Cellcycle	0,28	0,17	0,25	0,62	0,53
Church	0,26	0,20	0,16	0,63	0,57
Derisi	0,25	0,25	0,21	0,62	0,55
Eisen	0,21	0,21	0,19	0,64	0,54
Expr	0,17	0,14	0,15	0,68	0,48
Gasch1	0,17	0,24	0,18	0,76	0,69
Gasch2	0,15	0,17	0,16	0,63	0,55
Pheno	0,24	0,23	0,21	0,61	0,58
Seq	0,27	0,27	0,21	0,66	0,46
Spo	0,22	0,26	0,28	0,67	0,56

Rejeitada a hipótese nula e considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC foi estatisticamente superior ao MHC-ES nas três gerações (20, 60 e 100). Já comparando o resultado com o Clus-HSC esse foi estatisticamente superior apenas ao MHC-ES com 100 gerações.

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados do domínio Componente Celular.

Serão apresentados os resultados obtidos com 20, 60 e 100 gerações. As bases de dados selecionadas também foram as mesmas selecionadas anteriormente (Church e Derisi). A Tabela 39 apresenta os resultados obtidos nas duas bases de dados com a execução de 20 gerações. Observa-se que em todos os limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. O mesmo ocorre na execução com 60 gerações conforme mostra a Tabela 40 e na execução com 100 gerações como mostra a Tabela 41.

Comparando os resultados, mostrados nas Tabelas 39, 40 e 41, a Figura 68 apresenta os resultados da base de dados Church e os resultados obtidos na base de dados Derisi são mostrados na Figura 69.

Tabela 39 – Comparativo da Medida de Precisão e Revocação com 20 gerações na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,02	**1,00**	0,43	0,75	0,47	0,66	0,04	**0,85**	0,43	0,75	0,46	0,65
T(20)	0,02	**0,99**	0,51	0,68	0,57	0,57	0,06	**0,82**	0,51	0,68	0,54	0,56
T(30)	0,02	**0,95**	0,60	0,60	0,63	0,48	0,07	**0,78**	0,58	0,62	0,60	0,47
T(40)	0,03	**0,89**	0,63	0,56	0,71	0,41	0,13	**0,70**	0,63	0,53	0,66	0,42
T(50)	0,04	**0,79**	0,82	0,35	0,81	0,34	0,22	**0,58**	0,73	0,41	0,72	0,37
T(60)	0,05	**0,68**	0,89	0,31	0,85	0,32	0,31	**0,44**	0,91	0,28	0,85	0,29
T(70)	0,08	**0,54**	0,92	0,30	0,88	0,30	0,41	**0,19**	0,93	0,27	0,88	0,27
T(80)	0,13	**0,45**	0,93	0,28	0,89	0,28	0,43	**0,19**	0,93	0,27	0,89	0,26
T(90)	0,43	**0,32**	0,95	0,22	0,92	0,23	0,45	**0,19**	0,97	0,19	0,92	0,23
T(100)	0,64	**0,20**	0,99	0,11	0,96	0,11	0,49	**0,19**	1,00	0,10	0,96	0,17

Tabela 40 – Comparativo da Medida de Precisão e Revocação com 60 gerações na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,02	**0,99**	0,43	0,75	0,47	0,66	0,04	**0,84**	0,43	0,75	0,46	0,65
T(20)	0,02	**0,96**	0,51	0,68	0,57	0,57	0,06	**0,79**	0,51	0,68	0,54	0,56
T(30)	0,03	**0,90**	0,60	0,60	0,63	0,48	0,07	**0,77**	0,58	0,62	0,60	0,47
T(40)	0,04	**0,80**	0,63	0,56	0,71	0,41	0,10	**0,73**	0,63	0,53	0,66	0,42
T(50)	0,08	**0,59**	0,82	0,35	0,81	0,34	0,14	**0,72**	0,73	0,41	0,72	0,37
T(60)	0,19	**0,45**	0,89	0,31	0,85	0,32	0,19	**0,69**	0,91	0,28	0,85	0,29
T(70)	0,42	**0,19**	0,92	0,30	0,88	0,30	0,28	**0,52**	0,93	0,27	0,88	0,27
T(80)	0,49	**0,19**	0,93	0,28	0,89	0,28	0,27	**0,20**	0,93	0,27	0,89	0,26
T(90)	0,49	**0,19**	0,95	0,22	0,92	0,23	0,46	**0,19**	0,97	0,19	0,92	0,23
T(100)	0,49	**0,19**	0,99	0,11	0,96	0,11	0,49	**0,19**	1,00	0,10	0,96	0,17

Tabela 41 – Comparativo da Medida de Precisão e Revocação com 100 gerações na Ontologia Componente Celular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,02	**1,00**	0,02	1,00	0,02	1,00	0,02	**1,00**	0,02	1,00	0,02	1,00
T(10)	0,02	**0,99**	0,43	0,75	0,47	0,66	0,05	**0,86**	0,43	0,75	0,46	0,65
T(20)	0,02	**0,93**	0,51	0,68	0,57	0,57	0,06	**0,82**	0,51	0,68	0,54	0,56
T(30)	0,03	**0,86**	0,60	0,60	0,63	0,48	0,08	**0,76**	0,58	0,62	0,60	0,47
T(40)	0,06	**0,76**	0,63	0,56	0,71	0,41	0,13	**0,59**	0,63	0,53	0,66	0,42
T(50)	0,09	**0,46**	0,82	0,35	0,81	0,34	0,20	**0,49**	0,73	0,41	0,72	0,37
T(60)	0,20	**0,25**	0,89	0,31	0,85	0,32	0,22	**0,46**	0,91	0,28	0,85	0,29
T(70)	0,22	**0,20**	0,92	0,30	0,88	0,30	0,28	**0,31**	0,93	0,27	0,88	0,27
T(80)	0,28	**0,19**	0,93	0,28	0,89	0,28	0,26	**0,19**	0,93	0,27	0,89	0,26
T(90)	0,31	**0,19**	0,95	0,22	0,92	0,23	0,27	**0,19**	0,97	0,19	0,92	0,23
T(100)	0,49	**0,19**	0,99	0,11	0,96	0,11	0,49	**0,19**	1,00	0,10	0,96	0,17

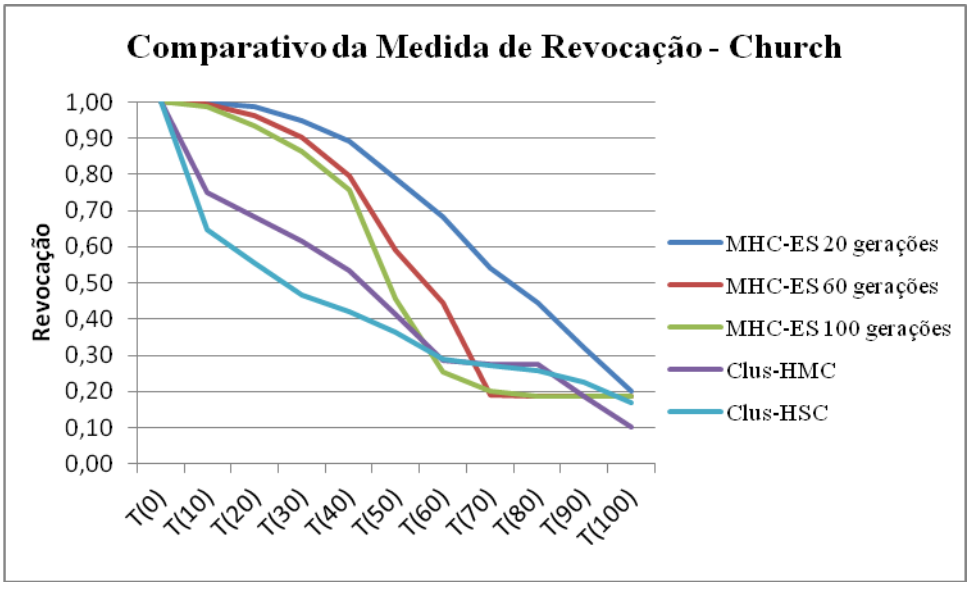


Figura 68 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Componente Celular.

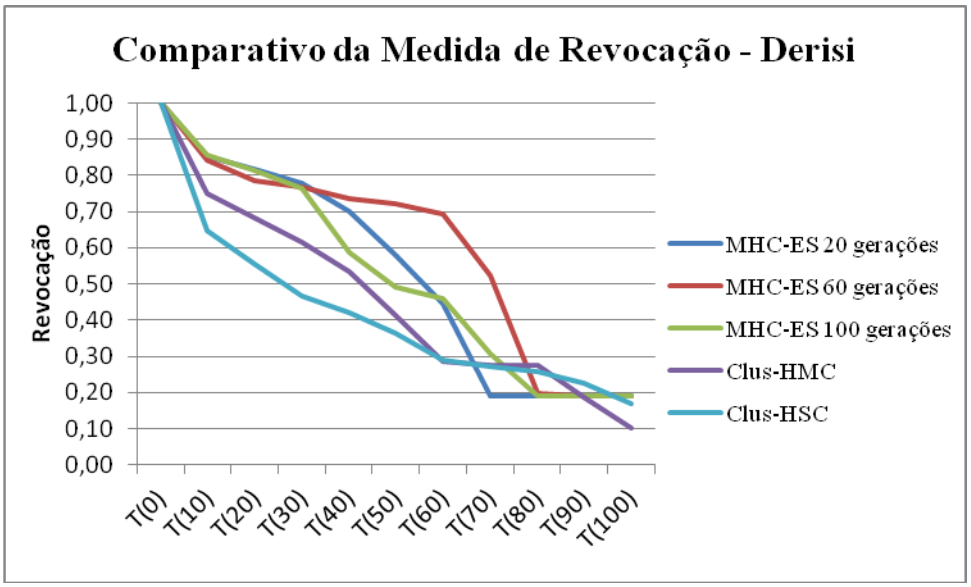


Figura 69 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Componente Celular.

Analisando a medida de revocação, em todas as bases de dados da ontologia Componente Celular e com todos os limiares, o algoritmo MHC-ES comparado como o Clus-HMC foi superior na maioria dos casos conforme mostra a Figura 70. A mesma comparação é feita entre o algoritmo MHC-ES e o Clus-HSC, conforme mostra a Figura 71.

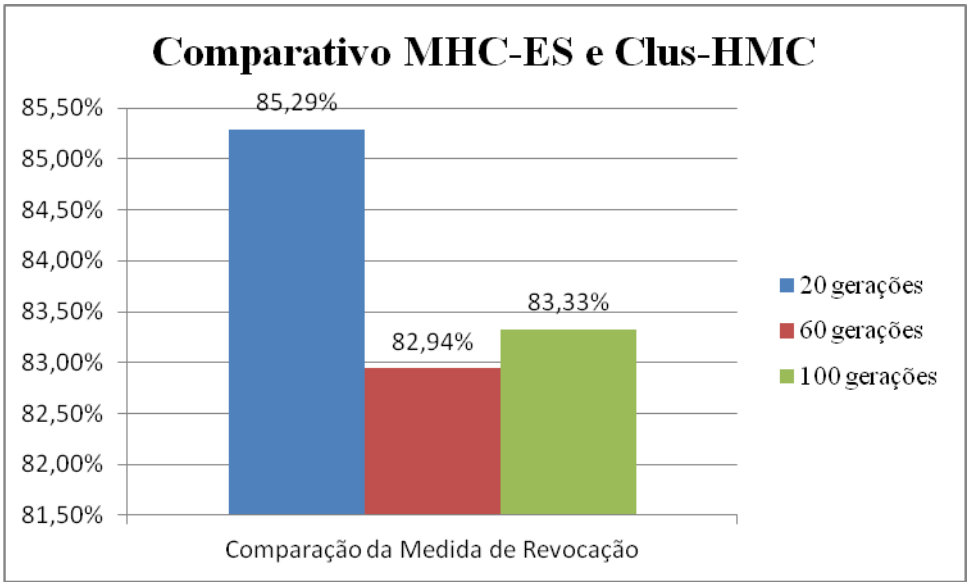


Figura 70 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HMC na Ontologia Componente Celular.

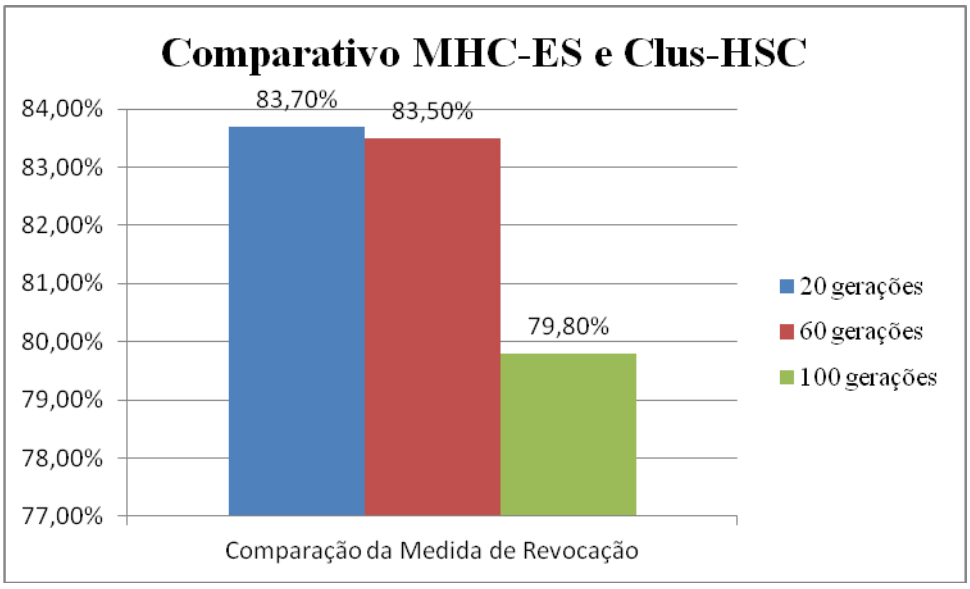


Figura 71 – Comparativo da Medida de Revocação entre o MHC-ES e o Clus-HSC na Ontologia Componente Celular.

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 20 gerações e o Clus-HMC, e substituindo os valores nas Equação 36 e Equação 37 têm-se $W^+ = 0$ e $W^- = 55$. Aplicando na Equação 38 tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$. Sendo assim, a hipótese nula deve ser rejeitada assumindo que a há diferença estatística entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 60 gerações e 100 gerações.

6.6 COMPARAÇÃO GERAL DOS RESULTADOS

Nesta Seção destacam-se os resultados gerais dos experimentos feitos nas bases de dados estruturadas em árvore (Funcat) e estruturadas em DAG (GO). Vale ressaltar que os experimentos realizados em estruturas na forma de árvore foram os experimentos iniciais em que apenas um rótulo da instância é avaliado, visto que tais bases utilizadas apresentam esta característica.

6.6.1 Resultados obtidos em estruturas em Árvore

Analisando os resultados apresentados no Apêndice A, observa-se que o algoritmo HC-CNN (BORGES & NIEVOLA, 2012a), (BORGES & NIEVOLA, 2012b), teve resultados significativos em ambas as medidas, distância e medida-Fh, chegando a alguns casos obter resultados superiores a 90%. No que se refere a quantidade de épocas, nota-se que houve pouca variação no resultado.

Já o algoritmo HC-ES teve resultados estatisticamente inferiores quando comparado ao HC-CNN. Esta observação fica clara ao observar a Tabela 45 e a Tabela 46 do Apêndice A.

Outros experimentos com o classificador HC-ES devem ser feitos, com parâmetros diferentes tal qual a quantidade de indivíduos e a quantidade de gerações a fim de verificar se o desempenho preditivo permanece o mesmo ou há melhoras significativas nos resultados.

Comparando ambos os classificadores desenvolvidos, HC-CNN e HC-ES, com o Clus-HSC através da medida AUCPR (**Erro! Fonte de referência não encontrada.**) notam-se resultados estatisticamente inferiores na maioria dos resultados exceto na execução do HC-CNN com 500 e 1000 épocas, que no teste de Friedman mostrou que não diferença estatística.

6.6.2 Resultados obtidos em estruturas DAG

Como se pode observar através dos experimentos realizados, com os classificadores desenvolvidos, o algoritmo MHC-CNN obteve resultados estatisticamente superiores ao MHC-ES.

Na base de dados completa, quando usado o algoritmo MHC-CNN, a medida de distância teve resultados estatisticamente melhores que a medida-Fh. Uma possível causa deve-se ao fato da utilização da medida de distância dependente de profundidade na qual foi atribuído pesos aos níveis da hierarquia. Assim, predições em níveis mais próximos da raiz tiveram um peso maior na avaliação, e as predições em níveis mais específicos pesos menores.

No que se refere à comparação dos resultados com o Clus-HMC e Clus-HSC, os dois algoritmos desenvolvidos (MHC-CNN e MHC-ES) tiveram desempenho preditivo estatisticamente inferior utilizando a curva *precision-recall*. Talvez, uma das possíveis causas do fato ocorrido foi adaptação dos classificadores, MHC-CNN e MHC-ES, para utilizar como

medida de avaliação AUCPR. Ambos os algoritmos utilizaram a medida de distância euclidiana, entre o neurônio de entrada com cada neurônio da camada de saída, visto que para a utilização da medida AUCPR a saída do classificador precisava ser um vetor de valores contínuos.

No que se refere aos resultados obtidos nas bases de dados com limiar igual a 300 nota-se que houve pouca melhora nos resultados comparados com a base de dados completa. Porém, devido à redução de instâncias houve um ganho no desempenho computacional.

Em se tratando das bases de dados separadas por ontologias observa-se que a ontologia Componente Celular teve resultados significativos principalmente quando utilizado a medida de distância. Uma possível causa desses resultados deve-se a características das bases de dados como mostrado na Tabela 10. Nota-se que as bases de dados desse domínio possuem uma quantidade de classes muito menor comparada com as outras ontologias. Além disso, há uma diminuição na quantidade de níveis da hierarquia e também na quantidade de classes que cada instância possui.

Nas ontologias Função Molecular e Processo Biológico o resultado preditivo foi inferior comparada a da ontologia Componente Celular. Uma das possíveis causas deve-se a características das bases de dados pertencentes as este domínio.

Porém, em se tratando do resultado obtido pelas medidas de avaliação a medida de distância também teve resultados estatisticamente melhores que a medida-Fh em ambos os classificadores, MHC-CNN e MHC-ES. No que se refere à comparação com o Clus-HMC e Clus-HSC, usando a media AUCPR, os resultados dos classificadores MHC-CNN e MHC-ES foram estatisticamente inferiores.

Analisando os resultados obtidos pela medida AUCPR observa-se que em todos os experimentos, nas diferentes bases de dados, essa teve resultado estatisticamente muito baixo. Porém, avaliando isoladamente cada medida precisão e revocação, medidas essas que dão origem a medida AUCPR, observa-se que em mais 90% dos resultados dos experimentos a medida de revocação, dos classificadores desenvolvidos, foi superior a medida de revocação do Clus-HMC e Clus-HSC.

7 CONCLUSÃO

A construção de um classificador hierárquico, multirrótulo, e utilizando a abordagem de classificação hierárquica global é apontada como uma das mais complexas abordagens. Isto porque algoritmos utilizados na classificação convencional não podem ser aplicados em problemas hierárquicos, na abordagem global, sem que haja modificações no algoritmo.

Esta tese apresentou um classificador usando a abordagem global para problemas de classificação hierárquica multirrótulo, utilizando uma rede neural artificial competitiva. O algoritmo proposto, denominado de MHC-CNN, treina uma rede neural competitiva baseada no algoritmo LVQ1.

Os resultados dos experimentos sugerem que o algoritmo MHC-CNN obtém desempenho preditivo competitivo quando utilizado as medidas de avaliação de distância, dependente de profundidade, e a hierárquica baseada na ancestralidade (medida-Fh). Esses resultados são encorajadores, visto que foi utilizado algoritmo LVQ1 convencional e que não foram feitas tentativas de otimização dos parâmetros de execução da rede neural.

Além disso, foi desenvolvido outro classificador, denominado de MHC-ES, que utiliza a estratégia evolucionária para o treinamento da rede neural.

Ademais, deve ser destacado que ambos os classificadores podem ser considerados os primeiros algoritmos que utilizam redes neurais para tratar de problemas hierárquicos estruturados em DAG e que utilizam a abordagem de classificação hierárquica global.

Comparando os resultados obtidos dos classificadores MHC-CNN e MHC-ES observa-se que o primeiro teve desempenho preditivo superior na maioria dos experimentos.

Houve grande dificuldade em comparar os resultados obtidos com outros algoritmos da literatura, pois não se encontrou nenhum que utilizasse as mesmas medidas de avaliação, medida de distância dependente de profundidade e medida-Fh, para que pudesse ser feita uma comparação equivalente, visto que essas foram as medidas selecionadas para avaliar o desempenho dos algoritmos propostos.

Assim, os classificadores desenvolvidos tiveram que ser adaptados para serem comparados com os algoritmos Clus-HMC e Clus-HSC, usando a medida de avaliação AUCPR.

Vale destacar que, além dos experimentos feitos nas bases de dados hierárquicas multirrótulo da GO, os experimentos iniciais foram feitos em bases de dados do Funcat estruturadas em árvore, em que uma instância apenas possui um rótulo/classe. Dessa maneira,

o primeiro classificador desenvolvido foi HC-CNN o qual tratava apenas de instâncias formadas por um único rótulo. Uma segunda versão do classificador HC-CNN foi denominada de HC-ES que utilizava a estratégia evolucionária para treinar a rede neural artificial.

Dessa maneira, pode-se dizer que este trabalho apresentou 4 versões de classificadores hierárquicos globais, sendo 2 deles para estrutura em árvore com instância possuindo apenas um único rótulo e os outros dois para problemas hierárquicos multirrótulos que podem ser utilizados em dados estruturados em DAG com em árvore.

7.1 TRABALHOS FUTUROS

Muitas pesquisas podem ser feitas envolvendo a classificação hierárquica multirrótulo estruturado em DAG. Primeiramente, no que se refere ao tratamento dos dados pode-se remover alguns níveis da hierarquia e como consequência algumas classes da hierarquia de classes. Isso irá diminuir a dificuldade que classificador ao predizer uma determinada classe. Como consequência disso, o custo computacional do classificador também irá diminuir proporcionalmente.

No que se refere ao classificador MHC-CNN, pode ser adaptada ao algoritmo às outras versões do algoritmo LVQ, tal qual a LVQ2, LVQ3, entre outras, e assim comparar os resultados, visto que na literatura sabe-se que essas versões apresentam resultados melhores na classificação convencional. Por isso, vale a pena testar na classificação hierárquica.

Outros experimentos podem ser feitos otimizando os parâmetros da rede neural, como a taxa de aprendizagem e a quantidade de épocas.

Uma versão local do classificador pode ser desenvolvida. Isso poderá facilitar na comparação do resultado do algoritmo selecionado como base, neste caso o LVQ, verificando assim o desempenho em ambas as abordagens: local e global.

Tratando do classificador MHC-ES, o qual apresentou resultados inferiores ao MHC-CNN, podem ser feitos outros experimentos com maior quantidade de indivíduos e gerações e assim também verificar o seu desempenho preditivo.

Em se tratando da avaliação feita na medida de revocação obtida a partir da medida AUCPR, cujos resultados foram na maioria dos casos muito superiores ao Clus, um estudo pode ser feito para avaliar se a curva ROC não fornece melhores resultados que a curva PR, visto que a medida AUCPR tem sido utilizada frequentemente para avaliar os resultados dos

classificadores que têm sido desenvolvidos, porém sem um consenso comum de que esta seja uma boa medida.

REFERÊNCIAS

ALEKSOVKI, D.; KOCEV, D.; DZEROSKI, S. Evaluation of distance measures for hierarchical multi-label classification in functional genomics. In Proc. of the 1st Workshop on Learning from Multi-Label Data (MLD). p. 5-16. 2009.

ALVES, R. T.; DELGADO, M. R., FREITAS, A. A. Multi-label hierarchical classification of protein functions with artificial immune systems. In Proc. Advances in Bioinformatics and Computational Biology. v. 5167, p.1-12. 2008.

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

ASTIKAINEN, K.; HOLMAND, L.; PITHANEN, E.; SZEDMAK, S.; ROUSU, J. Towards structured output prediction of enzyme function. **BMC** Proc 2(Suppl 4). 2008.

BÄCK, T.; FOGEL, D.B.; MICHALEWICZ, Z. **Evolutionary computation 2: advanced algorithms and operators**. Bristol and Philadelphia: Institute of Physics Publishing, 2000.

BARBEDO, J. G. A.; LOPES, A. Automatic genre classification of musical signals. EURASIP Journal on Advances in Signal Processing. v. 2007, n. 1, p. 157-168. 2007.

BALDI, P.; POLLASTRI, G. A Machine-Learning Strategy for Protein Analysis. **IEEE Intelligent Systems**. v.17, n. 2, p.28-35. 2002.

BARUTCUOGLU, Z.; SCHAPIRE, R. E.; TROYANSKAYA, O. G. Hierarchical multi-label prediction of gene function. **Bioinformatics**. v. 22 n. 7, p. 830-836. 2006.

BENNETT, P. N.; NGUYEN, N. Refined experts: improving classification in large taxonomies. In Proc. of the 32nd Int. ACM SIGIR conf. on Research and development in information retrieval, Boston, MA, USA. p. 11-18. 2009.

BRECHEISEN, S.; KRIEGEL, H. P.; KUNATH P.; PRYAKHIN, A. Hierarchical genre classification for large music collections. In: Proceedings of the IEEE 7th international conference on Multimedia&Expo. p. 1385–1388. 2006.

BI, W.; KWOK, J. T. Multi-label classification on tree- and DAG-structured hierarchies. Proceedings of the Twenty-Eighth International Conference on Machine Learning (ICML), Bellevue, WA, USA, June 2011.

BINDER, A.; KAWANABE, M.; BREFELD, U. Efficient classification of images with taxonomies. In: Proceedings of the 9th Asian conference on computer vision. 2009.

BLOCKELL, H.; BRUYNOOGHE, M.; DZEROSKI, S.; RAMON, J.; STRUYF, J. Hierarchical multi-classification. In Proc. of the First SIGKDD Workshop on Multi-Relational Data Mining (MRDM-2002), Edmonton, Canada, July. p. 21-35. 2002.

BLOCKELL, H.; SCHIETGAT, L.; STRUYF, J.; DZEROSKI, S.; CLARE, A. Decision trees for hierarchical multilabel classification: A case study in functional genomics. In Proc. of the

10th European Conf. on Principles and Practice of Knowledge Discovery in Databases. p. 18-29. 2006.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Hierarchical Classification using a Competitive Neural Network. In: 8th International Conference on Natural Computation (ICNC'12), 2012, Chongqing, China. 8th International Conference on Natural Computation (ICNC'12). Piscataway, NJ: IEEE Press, 2012. v. 1. p. 172-177.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Hierarchical Classification Using a Competitive Neural Network for Protein Function Prediction. In: 14th International Conference on Artificial Intelligence (ICAI'12), 2012, Las Vegas. 14th International Conference on Artificial Intelligence (ICAI'12). USA: CSREA Press, 2012. v. 1. p. 1-7.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Multi-Label Hierarchical Classification using a Competitive Neural Network for Protein Function Prediction. In: 2012 International Joint Conference on Neural Networks (IJCNN 2012), 2012, Brisbane, Austrália. 2012 International Joint Conference on Neural Networks (IJCNN 2012). Piscataway, NJ: IEEE Press, 2012. v. 1. p. 1-8.

BORGES, Helyane Bronoski; NIEVOLA, Julio Cesar. Classificador Hierárquico Multi-Classe usando uma Rede Neural Competitiva. In: XIX SEMIC. Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação, 2011, Curitiba. Caderno de Resumos do XIX Seminário de Iniciação Científica. XIII Mostra de Pesquisa da Pós-Graduação. Curitiba: Champagnat, 2011. v. 1. p. 137-137.

BORGES, Helyane Bronoski; NIEVOLA, J. C. Classificador Hierárquico Multiclasses usando uma Rede Neural Competitiva para Predição Funcional de Proteínas. In: XVIII SEMIC. Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação, 2010, Curitiba. Caderno de Resumos do XVIII Seminário de Iniciação Científica. XII Mostra de Pesquisa da Pós-Graduação. Curitiba: Champagnat, 2010. v. 1. p. 281-281.

BURRED, J. J.; LERCH, A. A hierarchical approach to automatic musical genre classification. In Proc. of the 6th Int. Conf. on Digital Audio Effects, London, UK, Set. p. 8-11. 2003.

CAI, L.; HOFMANN, T. Hierarchical document categorization with support vector machines. In: Proceedings of the 13th ACM international conference on information and knowledge management, p. 78–87. 2004.

CAI, L.; HOFMANN, T. Exploiting known taxonomies in learning overlapping concepts. In: Proceedings of the 20th international joint conference on artificial intelligence. p. 714-719. 2007.

CESA-BIANCHI, N.; GENTILE, C. ZANIBONI, L. Hierarchical classification: combining Bayes with SVM. In: Proceedings of the 23rd international conference on machine learning. p. 177–184, 2006a.

CESA-BIANCHI, N.; GENTILE, C. ZANIBONI, L (2006b) Incremental algorithms for hierarchical classification. **Journal of Machine Learning Research**. v. 7, p. 31–54, 2006b.

CESA-BIANCHI, N.; VALENTINI, G. Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. In: Third international workshop on machine learning in systems biology. JMLR: Workshop and Conference Proceedings. v. 8, p. 14-29, 2009.

CERRI, R.; CARVALHO, A. C. P. L. F. Hierarchical Multilabel Protein Function Prediction Using Local Neural Networks. In Brazilian Symposium on Bioinformatics. v. 6832, p. 10-17, 2011.

CERRI, R.; CARVALHO, A. C. P. L. F. Hierarchical multilabel classification using topdown label combination and artificial neural networks. In: Brazilian Symposium on Artificial Neural Networks. p. 253–258, 2010.

CHAKRABARTI, S.; DOM, B.; AGRAWAL, R.; RAGHAVAN, P. Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. **The VLDB Journal.** v. 7, p. 163-178, 1998.

CHU, S.; Derisi, J.; Eisen, M.; Mulholland, J. Botstein, D.; Brown, P. O.; Herskowitz, I. The transcriptional program of sporulation in budding yeast. **Science**, v. 282, p. 699-705, 1998.

CLARE, A., KING, R. D. Knowledge discovery in multi-label phenotype data. In Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery and Data Mining (PKDD-2001), Freiburg, Germany. p. 42-53. 2001.

CLARE, A., KING, R. D. Predicting gene function in Saccharomyces Cerevisiae. **Bioinformatics**. v.19, s. 2, p. 42-49, Out. 2003.

CLARE, A. **Machine Learning and Data Mining for Yeast Functional Genomics**. Tese (Doutorado) - University of Wales, Aberystwyth, Wales, 2003.

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

DARWIN, C. **On the origin of species by means of natural selection or the preservation of favored races in the struggle for life**. Murray, London, UK. 1859.

DAVIS, J., GOADRICH, M. The relationship between precision-recall and roc curves. In: International Conference on Machine Learning. p. 233–240, 2006.

D'ALESSIO, S.; MURRAY, K. Schiaffino R, Kershenbaum, A. The effect of using hierarchical classifiers in text categorization. In: Proceedings of the 6th international conference Recherche d´ Information Assistee par Ordinateur, p. 302-313. 2000.

DeCORO, C.; BARUTCUOGLU, Z.; FIEBRINK, R. Bayesiana aggregation for hierarchical gene classification. In: Proc. of the 8th Int. Conf. on Music Information Retrieval, Vienna, Austria. p. 77-80. 2007.

DEKEL, O.; KESHET, J.; SINGER, Y. Large margin hierarchical classification. In: Proceedings of the 21th international conference on Machine learning. 2004a.

DEKEL, O.; KESHET, J.; SINGER, Y. An online algorithm for hierarchical phoneme classification. In: Proceedings of the 1st machine learning for multimodal interaction workshop. Lecture notes in computer science, vol 3361. Springer, Berlin, p. 146-158. 2004b.

DERISI, J.; IYER, V.; BROWN, P. Exploring the metabolic and genetic control of gene expression on a genomic scale. **Science**. v. 278, p. 680-686. 1997.

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

DIMITROVSKI, I.; KOCEV, D.; LOSKOVSKA, S.; DZEROSKI, S. Hierarchical annotation of medical images. In Proc. of the 11th Int. Multiconference Information Society. v. A, p. 174-177. 2008.

DUMAIS, S.; CHEN, H. Hierarchical classification of web content. Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval. p. 256–263, 2000.

EISEN, M.; SPELLMAN, P.; BROWN, P.; BOTSTEIN, D. Cluster analysis and display of genome-wide expression patterns. In: Proceedings of the National Academy of Sciences. v. 95, p. 14863-14868. 1998.

ESULI, A; FAGNI, T; SEBASTIANI, F.Boosting multi-label hierarchical text categorization. Information Retrieval, v. 11, n. 4. p. 287-313. 2008.

FAWCETT, T. An introduction to ROC analysis. **Pattern Recognition Letters**, v. 227, n. 8, p. 861-874, June, 2006.

FREITAS, A. A.; CARVALHO, A. C. P. F. A Tutorial on Hierarchical Classification with Applications in Bioinformatics. In: Taniar, D. Research and Trends in Data Mining Technologies and Applications. Advances in Data Warehousing and Mining. Hershey, PA, USA: IGI Publishing, 2007. Cap.7, p. 179-209.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. **Journal of the American Statistical Association**. v. 32, p. 675–701, 1937.

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

GASCH, A. P.; HUANG, M.; METZNER, S.; BOTSTEIN, D.; ELLEDGE, S. J.; BROWN, P.O. Genomic expression responses to dna-damaging agents and the regulatory role of the yeast atr homolog mec1p. **Molecular Biology of the Cell**. v. 12, n. 10, p. 2987-3000. 2001.

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

GAUCH, S.; CHANDRAMOULI, A.; RANGANATHAN, S. Training a hierarchical classifier using inter document relationships. **Journal of the American Society for Information Science and Technology**. v. 60. p. 47-58. 2009.

GENE ONOTOLOGY. The Gene Ontology. 1998. Disponível em: http://www.geneontology.org/GO. Acessado em: 11/04/2010.

GUAN, Y.; MYERS, C. L.; HESS, D. C. BARUTCUOGLU, Z.; CAUDY, A. A.; TROYANSKAYA, O. G. Predicting gene function in a hierarchical context with an ensemble of classifiers. **Genome Biology**, v. 9, p. 1-18. June, 2008.

HAMED, H. N. B. A. **Particle Swarm Optimization for Neural Network Learning Enhancement**. Malaysia, 2006. 83 p. Dissertação (Mestrado) - Faculty of Computer Science and Information System, Universiti Teknologi Malaysia, Malaysia, 2006.

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

HAYETE, B; BIENKOWSKA, J. Gotrees: predicting go associations from protein domain composition using decision trees. In: Proceedings of the Pacific symposium on biocomputing, p. 127–138. 2005.

HOLDEN, N.; FREITAS, A. A. Hierarchical classification of protein function with ensembles of rules and particle swarm optimisation. **Applied Soft Computing.** v. 13, p. 259–272. 2009.

HOLDEN, N.; FREITAS, A. A. Improving the performance of hierarchical classification with swarm intelligence. In Proceedings of the 6th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBio 2008), LNCS 973, p. 48–60. 2008.

HOLDEN, N.; FREITAS, A. A. A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data. In Proceedings of the 2005 IEEE Swarm Intelligence Symposium, Pasadena, California. p. 100-107. 2005.

HOLDEN, N.; FREITAS, A. A. Hierarchical classification of G-Protein-Coupled Receptors with a PSO/ACO Algorithm. In Proceedings of the 2006 IEEE Swarm Intelligence Symposium, Indianapolis, Indiana, USA, Set. p. 77-84. 2006.

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

IPEIROTIS, P., G.; GRAVANO, L.; SAHAMI, M. Probe, count, and classify: categorizing hidden web databases. **SIGMOD**, v. 30, n. 2, p. 67-78, May, 2001.

JENSEN, L. J.; GUPTA, R.; BLOM, N.; DEVOS, D.; TAMAMES, J.; KESMIR, C.; NIELSEN, H.; STAERFELDT, H. H.; RAPACKI, K.; WORKMAN, C.; ANDERSEN, C. A.; KNUDENS, S.; KROGH, A.; VALENCIA, A.; BRUNAKI, S. Prediction of human protein function from post-translational modifications and localization features. **Journal of Molecular Biology**. v. 319, n. 5, p. 1257-1265. 2002.

JENSEN, L. J.; GUPTA, R.; STAERFELD, H. H.; BRUNAK, S. Prediction of human protein function according to Gene Ontology categories. **Bioinformatics**. v. 19, n. 5, p. 635–642. 2003.

JIN, B.; MULLER, B.; ZHAI, C.; LU, X. Multi-label literature classification based on the gene ontology graph. **BMC Bioinformatics.** v. 9, n. 525, p. 1-15. June, 2008.

KING, O. D.; FOULGER, R. E.; DWIGHT, S. S. WHITE, J. V. ROTH, F. Predicting gene function from patterns of annotation. **Genome Research**. v. 13, p. 896-904. 2003.

KIRITCHENKO, S.; MATWIN, S.; FAMILI, A. F. Hierarchical Text Categorization as a Tool of Associating Genes with Gene Ontology Codes. In Proceedings of the 2nd European Workshop on Data Mining and Text Mining for Bioinformatics, Pisa, Italy. p. 26-30. 2004.

KIRITCHENKO, S. **Hierarchical Text Categorization and Its Application to Bioinformatics**. Ottawa, 2005, 187 p. Tese (Doutorado) - Information Technology and Engineering, University of Ottawa, Ottawa, Canadá, 2005.

KIRITCHENKO, S.; MATWIN, S.; FAMILI, A. F. Functional annotation of genes using hierarchical text categorization. In: Proceedings of the ACL workshop on linking biological literature, ontologies and databases: mining biological semantics. 2005.

KIRITCHENKO, S.; MATWIN, S.; NOCK, R.; FAMILI, A. F. Learning and evaluation in the presence of class hierarchies: Application to text categorization. In Proc. of the 19th Canadian Conf. on Artificial Intelligence, Lecture Notes in Artificial Intelligence. v. 4013, p. 395-406. 2006.

KOHONEN, T. The Self-Organizing Map. **Proceedings of IEEE**. v. 78, n. 9. p. 1464-1480. 1990.

KOHONEN, T.; HYNNINEN, J.; KANGAS, J.; LAAKSONEN, J.; TORKKOLA, J. LVQ_PAK: The Learning Vector Quantization Program Package, Version 3.1, Abr, 1995. Disponível em: www.cis.hut.fi/research/papers/lvq_tr96.ps/ - Acessado em: 20/10/2009.

KOLLER, D.; SAHAMI, M. Hierarchically classifying documents using very few words. In Proc. of the 14th Int. Conf. on Machine Learning (ICML 1997), San Francisco, CA, USA, July, p. 170-178. 1997.

KUMAR, A. et al. Triples: A database of gene function in s. cerevisiae. **Nucleic Acids Research**. v. 28, p. 81-84, 2000.

LABROU, Y.; FININ, T. Yahoo! as an ontology - using yahoo! categories to describe documents. In: Proceedings of the ACM conference on information and knowledge management. p. 180-187. 1999.

LAEGREID, A.; HVIDSTEN, T. R.; MIDELFART, H. Predicting gene ontology biological process from temporal gene expression patterns. **Genome Research**. v. 13, p. 965-979. 2003.

LESK, A. M. **Introdução à Bioinformática**. 2ª ed. Porto Alegre: Artmed, 2008.

LI, T.; OGIHARA, M. Music genre classification with taxonomy. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, pp 197-200. 2005.

MCCALLUM, A.; ROSENFELD, R.; MICHELL, T. M.; NG, A. Y. Improving text classification by shrinkage in a hierarchy of classes. In: Proceedings of the international conference on machine learning. p. 359-367. 1998.

MCKAY, C.; FUJINAGA, I. Automatic genre classification using large high-level musical feature sets. In: Proceedings of the international conference on music information retrieval, p. 525-530. 2004.

MEWES, H. W.;FRISHMAN, D.; GÜLDENER, U.; MANNHAUPT, G.; MAYER, K.; MOKREJS, M.; MORGENSTERN, B.; MÜNSTERKÖTTER, M.; RUDD, S.; WEIL, B. Mips: A database for protein sequences and complete genomes. **Nucleic Acids Research**. v. 27, p. 44-48, 1999.

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

PRATI, C. R.; BATISTA, A. P. A.; MONARD, M. C. Curva ROC para avaliação de Classificadores. **IEEE Latin America Transactions.** v. 6, n. 2, Junho, 2008.

OLIVER, S. A network approach to the systematic analysis of yeast gene function. **Trends in Genetics**. v. 12, n. 7, p. 241–242. 1996.

OTERO, F. E. B.; FREITAS, A. A.; JOHNSON, C. G. A hierarchical classification ant colony algorithm for predicting gene ontology terms. In: Pizzuti C, Ritchie M, GiacobiniM(eds) Proceedings of the 7th European conference on evolutionary computation, machine learning and data mining in bioinformatics (EvoBio). Lecture Notes in Computer Science, v. 5483. Springer, Berlin, p. 68–79. 2009.

OTERO, F. E. B.; FREITAS, A. A.; JOHNSON, C. G. A hierarchical multi-label classification ant colony algorithm for protein function prediction. **Memetic Computing.** v. 2, p. 165–181, 2010.

PENG, X.; CHOI, B. Document classifications based on word semantic hierarchies. In: Proceedings of the international conference on artificial intelligence and applications. p. 362–367. 2005.

QIU, X.; GAO, W.; HUANG, X. Hierarchical multi-class text categorization with global margin maximization. In: Proceedings of the Joint conference of the 47th Annual Meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP, Association for computational linguistics. P. 165-168. 2009.

RECHENBERG, I. Cybernetic Solution Path of an Experimental Problem. Roy. Aircr. Establ. Libr. Transl., 1122, Farnborough, Hants, UK.1965.

RECHENBERG, I. Evolutionsstrategic: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. **Frommann-Holzboog Verlag Stuttgart**, 1973.

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

ROUSU, J.; SAUNDERS, C.; SZEDMAK, S.; SHAWE-TAYLOR, J. Learning hierarchical multi-category text classification models. In: Proceedings of the 22nd international conference on machine learning. p. 744–751. 2005.

ROUSU, J.; SAUNDERS, C.; SZEDMAK, S.; SHAWE-TAYLOR, J. Kernel-based learning of hierarchical multilabel classification models. **Journal of Machine Learning Research**. v. 7, p. 1601-1626. 2006.

ROTH, F. P. HUGHES, J. D.; ESTEP, P.W. CHURCH, G. M. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. **Nature Biotechnology**. v. 16, p. 939-945. 1998.

RUIZ, M..; SRINIVASAN, P. Hierarchical text categorization using neural networks. **Information Retrieval**. v. 5, p. 87–118, 2002.

SECKER, A.; DAVIES, M.; FREITAS, A.; TIMMIS, J.; MENDAO, M.; FLOWER, D. An experimental comparison of classification algorithms for the hierarchical prediction of protein function. **Expert Update (Magazine of the British Computer Society's Specialist Group on AI**. v. 9, n. 3, p. 17-22. 2007.

SEEGER, M. W. Cross-validation optimization for large scale structured classification kernel methods. **Journal of Machine Learning Research**. v. 9, p. 1147-1178. 2008.

SCHWEFEL, H. P. Evolutionsstrategie und numerische optimierung. Dissertação (Mestrado) - Technische Universitat Berlin, 1975.

SEIFFERT, U.; HAMMER, B.; KASKI, S.; VILLMANN, T. Neural networks and machine learning in bioinformatics - theory and applications. In European Symposium on Artificial Neural Networks, Bruges, Belgium. p..521–532. 2006.

SILLA JR C. N., FREITAS, A. A. A global-model naive bayes approach to the hierarchical prediction of protein functions. In: Proceedings of the 9th IEEE international conference on data mining, p. 992–997. 2009a.

SILLA JR C. N., FREITAS, A. A. Novel top-down approaches for hierarchical classification and their application to automatic music genre classification. Int. Conf. on Systems, Man and Cybernetics (SMC 2009). San Antonio, USA, Oct. p. 3599-3604. 2009b.

SILLA JR C. N., FREITAS, A. A. A survey of hierarchical classification across different application domains. **Data Mining and Knowledge Discovery**. Abr, 2010.

SPELLMAN, P. et al. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. **Molecular Biology of the Cell**. v. 9, p. 3273-3297. 1998.

SUN, A.; LIM, E. Hierarchical Text Classification and Evaluation. In Proceedings of the International Conference on Data Mining (ICDM 2001), California, USA, p. 521-528. 2001.

SUN, A.; LIM, E.-P.; NG, W.-K. Performance measurement framework for hierarchical text classification. **Journal of the American Society for Information Science and Technology.** v. 54, n. 11, p. 1014–1028, 2003.

TU, K.; YU, H.; GUO, Z.; LI, X. Learnability-based further prediction of gene functions in Gene Ontology. **Genomics**. v. 84, p. 922-928. 2004.

VALENTINI, G. True path rule hierarchical ensembles. In: Kittler J, Benediktsson J, Roli F (eds) Proc. of the Eighth Int. Workshop on Multiple Classifier Systems, Springer, Lecture Notes in Computer Science. v. 5519, p. 232-241. 2009.

VALENTINI, G.; RE, M. Weighted true path rule: a multi-label hierarchical algorithm for gene function prediction. In: Proceedings of the 1st workshop on learning from multi-label data (MLD) held in conjunction with ECML/PKDD. p. 132–145, 2009.

VENS, C.; STRUYF, J.; SCHIETGAT, L.; DZEROSKIi, S.; BLOCKEEL, H. Decision trees for hierarchical multi-label classification. **Machine Learning**. v. 73, n. 2, p. 185-214. 2008.

XIAO, Z.; DELLANDRÉA, E.; DOU, W.; CHEN, L Hierarchical Classification of Emotional Speech. ISMW '07 Proceedings of the Ninth IEEE International Symposium on Multimedia Workshops. p. 291-296. 2007.

XUE, G.; XING, D.; YANG, Q.; YU, Y. Deep classification in large-scale text hierarchies. In Proc. of the 31st annual int. ACM SIGIR conf. on Research and development in information retrieval, Singapore, July, p. 619-626. 2008.

WANG, K.; ZHOU, S.; LIEW, S. C. Building hierarchical classifiers using class proximity. In: In Proceedings of the 25th conference on very large data base. Morgan Kaufmann Publishers, San Francisco. p. 363-374. 1999.

WANG, J.; SHEN, X.; PAN, W. On Large margin hierarchical classification with multiple paths. **Journal of the American Statistical Association**. v. 104, n. 487, p. 1213–1223. 2009.

WEINERT, W. R., LOPES, H. S. Neural networks for protein classification. **Applied Bioinformatics**. v. 3, n. 1, p. 41-48. 2004.

WEIGENED, A. S.; WIENER, E. D. PEDERSEN, J. O. Exploiting hierarchy in text categorization. **Information Retrieval**. v.1. p. 193–216, 1999.

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

WITTEN I. H.; IAN H.; FRANK, E. **Data Mining: Practical machine learning tools and techniques**, 2nd Edition, San Francisco: Morgan Kaufmann, 2005.

APÊNDICE A – EXPERIMENTOS REALIZADOS COM BASES DE DADOS ESTRUTURADAS EM ÁRVORE

Neste Apêndice são apresentados os experimentos feitos em bases de dados estruturadas em árvores obtidas do Funcat. A Seção A.1 apresenta a metodologia dos experimentos. A Seção A.2 mostra os resultados obtidos com os algoritmos HC-CNN e HC-ES. E por fim, a Seção A.3 faz uma comparação entre os resultados obtidos.

A1 METODOLOGIA DOS EXPERIMENTOS

Como descrito anteriormente os experimentos foram feitos em 8 bases de dados do Funcat (HOLDEN & FREITAS, 2009). A Tabela 42 apresenta as características de cada base de dados. Além das características apresentadas é importante destacar que essas bases não possuem dados faltantes. Para a realização dos experimentos, as bases de dados foram normalizadas utilizando a técnica MinMax e divididas em 2/3 para o treinamento e 1/3 para o teste.

Tabela 42 – Características das bases de dados do Funcat.

Bases de Dados	Quant. Amostras	Quant. Atributos	Quant. Classes	Quant. Classes por Nível
ECinterproFinal	14036	1216	331	6/41/96/188
ECpfamFinal	13995	708	334	6/41/96/191
ECprintsFinal	14038	382	352	6/45/92/209
ECprositeFinal	14048	585	324	6/42/89/187
GPCRinterproFinal	7461	450	198	12/54/82/50
GPCRpfamFinal	7077	75	192	12/52/79/49
GPCRprintsFinal	5422	282	179	8/46/76/49
GPCRprositeFinal	6261	128	187	9/50/79/49

A2 RESULTADOS OBTIDOS

Foram feitos experimentos com os algoritmos HC-CNN (BORGES & NIEVOLA, 2012a) & (BORGES & NIEVOLA, 2012b) e HC-ES. Para a execução HC-CNN foi utilizado a mesma taxa de aprendizagem utilizada nos outros experimentos e com diferentes épocas. A Tabela 43 apresenta os resultados obtidos pelo HC-CNN com 50, 500 e 1000 épocas utilizando as medidas de distância e a Medida-Fh. A Tabela 44 apresenta os resultados obtidos pelo HC-ES com 20, 60 e 100 gerações.

Tabela 43 – Resultados obtidos pelo HC-CNN nas Bases de Dados do Funcat.

	50 épocas		500 épocas		1000 épocas	
Bases de Dados	**Distância**	**Fh**	**distância**	**Fh**	**distância**	**Fh**
ECinterproFinal	84,1%	81,3%	84,8%	82,9%	85,3%	83,4%
ECpfamFinal	85,4%	83,2%	85,5%	83,3%	85,7%	83,6%
ECprintsFinal	85,3%	83,6%	87,4%	85,4%	88,0%	86,3%
ECprositeFinal	89,8%	87,5%	91,6%	90,5%	91,8%	90,7%
GPCRinterproFinal	77,5%	70,4%	75,9%	69,0%	75,6%	68,0%
GPCRpfamFinal	85,4%	60,7%	67,0%	57,5%	68,8%	71,8%
GPCRprintsFinal	79,1%	71,6%	79,0%	71,1%	78,7%	71,8%
GPCRprositeFinal	51,5%	51,1%	47,8%	37,0%	55,3%	47,5%

Tabela 44 - Resultados obtidos pelo HC-ES nas Bases de Dados do Funcat.

	20 gerações		60 gerações		100 gerações	
Bases de Dados	**Distância**	**Fh**	**distância**	**Fh**	**distância**	**Fh**
ECinterproFinal	15,8%	15,9%	18,0%	15,9%	19,2%	17,1%
ECpfamFinal	18,5%	14,1%	17,1%	14,6%	17,5%	13,5%
ECprintsFinal	16,2%	15,8%	15,3%	13,7%	14,5%	11,5%
ECprositeFinal	16,1%	13,7%	18,2%	13,0%	16,4%	14,6%
GPCRinterproFinal	35,9%	25,9%	33,8%	26,6%	35,5%	25,6%
GPCRpfamFinal	36,8%	26,7%	43,7%	33,8%	43,1%	33,9%
GPCRprintsFinal	40,6%	40,0%	44,1%	38,0%	49,7%	44,4%
GPCRprositeFinal	44,1%	39,9%	48,1%	43,2%	48,0%	42,9%

A3 COMPARAÇÃO DOS RESULTADOS

A Tabela 45 mostra o comparativo dos dois algoritmos, HC-CNN e HC-ES, como apresentados na Seção A2 quando utilizado a medida de distância.

Tabela 45 – Comparação entre HC-CNN e HC-ES usando a Medida de Distância.

	HC-CNN			HC-ES		
	50	**500**	**1000**	**20**	**60**	**100**
ECinterproFinal	84,1%	84,8%	85,3%	15,8%	18,0%	19,2%
ECpfamFinal	85,4%	85,5%	85,7%	18,5%	17,1%	17,5%
ECprintsFinal	85,3%	87,4%	88,0%	16,2%	15,3%	14,5%
ECprositeFinal	89,8%	91,6%	91,8%	16,1%	18,2%	16,4%
GPCRinterproFinal	77,5%	75,9%	75,6%	35,9%	33,8%	35,5%
GPCRpfamFinal	85,4%	67,0%	68,8%	36,8%	43,7%	43,1%
GPCRprintsFinal	79,1%	79,0%	78,7%	40,6%	44,1%	49,7%
GPCRprositeFinal	51,5%	47,8%	55,3%	44,1%	48,1%	48,0%

Calculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 28{,}79$ e $F_F = 18{,}0$. Conforme o valor tabelado para os graus de liberdade 7 e 35 na distribuição F de Snedecor encontra-se o valor crítico de $F(7;35) = 2{,}29$. Desta maneira a $F_F > F(7;35)$ logo a hipótese nula pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}67$. Considerando os dados da Tabela 45, o algoritmo HC-CNN com 1000 épocas é estatisticamente superior aos resultados do algoritmo HC-ES também nos três casos selecionados: 20, 60 e 100 gerações. Além disso, o algoritmo HC-CNN com 50 e 500 épocas é estatisticamente superior ao HC-ES na execução com 20 gerações.

Já a Tabela 46 mostra o comparativo dos dois algoritmos, HC-CNN e HC-ES utilizando a medida-Fh.

Tabela 46 – Comparação entre HC-CNN e HC-ES usando a Medida-Fh.

	HC-CNN (em épocas)			**HC-ES (em gerações)**		
	50	**500**	**1000**	**20**	**60**	**100**
ECinterproFinal	81,3%	82,9%	83,4%	15,9%	15,9%	17,1%
ECpfamFinal	83,2%	83,3%	83,6%	14,1%	14,6%	13,5%
ECprintsFinal	83,6%	85,4%	86,3%	15,8%	13,7%	11,5%
ECprositeFinal	87,5%	90,5%	90,7%	13,7%	13,0%	14,6%
GPCRinterproFinal	70,4%	69,0%	68,0%	25,9%	26,6%	25,6%
GPCRpfamFinal	60,7%	57,5%	71,8%	26,7%	33,8%	33,9%
GPCRprintsFinal	71,6%	71,1%	71,8%	40,0%	38,0%	44,4%
GPCRprositeFinal	51,1%	37,0%	47,5%	39,9%	43,2%	42,9%

Calculando os valores de $X_F^2 = 28{,}27$ e $F_F = 16{,}9$ utilizados no teste de Friedman e conforme o valor tabelado para os graus de liberdade 7 e 35 na distribuição F de Snedecor encontra-se o valor crítico de $F(7;35) = 2{,}29$. Deste modo a $F_F > F(7;35)$ logo a hipótese nula pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}67$. Considerando os dados da Tabela 45, o algoritmo HC-CNN, com 1000 épocas, é estatisticamente superior aos resultados do algoritmo HC-ES nos três casos selecionados: 20, 60 e 100 gerações. Já o HC-CNN, com 50 épocas é estatisticamente superior ao algoritmo HC-ES com a execução de 20 gerações.

APÊNDICE B – RESULTADOS OBTIDOS NAS BASES DE DADOS SEPARADAS POR ONTOLOGIAS

B1 ONTOLOGIA FUNÇÃO MOLECULAR

Nesta Seção são apresentados os comparativos realizados, na Ontologia Função Molecular, entre os algoritmos: MHC-CNN x MHC-ES, MHC-CNN x Clus (HMC e HSC) e MHC-ES x Clus (HMC e HSC).

B1.1 Comparação MHC-CNN e MHC-ES

A Tabela 47 mostra os resultados dos algoritmos MHC-CNN e MHC-ES na base de dados da ontologia Função Molecular, quando utilizado a medida de distância.

Tabela 47 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida de distância na Ontologia Função Molecular.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	55,4%	55,8%	55,9%	45,7%	46,8%	47,1%
Church	53,8%	54,8%	54,8%	57,3%	58,3%	57,5%
Derisi	52,6%	53,7%	53,5%	55,6%	56,1%	55,6%
Eisen	55,6%	55,7%	55,8%	46,9%	46,5%	45,6%
Expr	62,5%	62,7%	63,0%	50,9%	54,0%	51,5%
Gasch1	56,5%	57,6%	58,5%	50,0%	55,5%	50,8%
Gasch2	56,3%	56,8%	57,3%	49,0%	47,9%	59,8%
Pheno	53,1%	53,1%	52,7%	59,2%	57,7%	56,4%
Seq	62,7%	64,1%	65,0%	56,7%	56,2%	56,4%
Spo	56,9%	57,1%	57,4%	57,9%	58,0%	57,9%

Calculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 4,.47$ e $F_F = 0,9$. Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2,09$. Deste modo a hipótese nula não é rejeitada, $F_F < F(9;45)$ indicando que não há diferença significativa entre os resultados dos algoritmos.

A Tabela 48 mostra os resultados dos dois algoritmos, quando utilizado a medida-Fh.

Tabela 48 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida-Fh na Ontologia Função Molecular.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	45,1%	42,2%	41,9%	25,8%	29,7%	26,7%
Church	43,5%	45,2%	42,5%	30,7%	28,4%	30,6%
Derisi	36,3%	36,7%	35,9%	29,9%	28,2%	29,7%
Eisen	36,3%	40,4%	39,8%	30,2%	24,6%	24,5%
Expr	51,4%	52,1%	48,7%	23,1%	26,7%	24,1%
Gasch1	46,6%	41,6%	39,4%	27,8%	30,2%	32,2%
Gasch2	39,6%	41,5%	40,1%	31,8%	30,1%	25,1%
Pheno	35,0%	42,3%	41,5%	22,4%	29,5%	24,0%
Seq	54,8%	53,1%	53,4%	29,6%	29,1%	30,2%
Spo	35,6%	34,9%	34,5%	27,6%	27,5%	29,0%

Calculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 40{,}4$ e $F_F = 37{,}9$. Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2{,}09$. Deste modo a $F_F > F(9;45)$ logo a hipótese nula pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}38$. Considerando os dados da Tabela 48, o algoritmo MHC-CNN nas três épocas selecionadas (50, 500 e 1000), é estatisticamente superior ao algoritmo MHC-ES nos casos da seleção de 20 e 60. Além disso, o MHC-CNN, com 50 e 500 épocas, é estatisticamente superior ao resultado do algoritmo MHC-ES também com a seleção de 100 indivíduos.

B1.2 Comparação MHC-CNN e Clus

A Tabela 49 mostra os resultados dos algoritmos utilizando a medida AUPRC com 50, 500 e 1000 épocas.

Calculando os valores para as equações utilizadas no teste de Friedman têm-se $X_F^2 = 32{,}5$ e $F_F = 39$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo

Clus-HMC foi estatisticamente superior ao MHC-CNN nas três épocas (50, 500 e 1000). Já o Clus-HSC foi estatisticamente superior ao MHC-CNN na execução com 50 e 1000 épocas.

Tabela 49 – Comparativo da Medida AUPRC entre o MHC-CNN e o Clus-HMC e Clus-HSC na Ontologia Função Molecular.

	MHC-CNN (em épocas)			Clus	
	50	**500**	**1000**	**HMC**	**HSC**
Cellcycle	0,09	0,08	0,08	0,34	0,27
Church	0,10	0,10	0,10	0,36	0,34
Derisi	0,08	0,10	0,10	0,34	0,29
Eisen	0,08	0,08	0,07	0,34	0,28
Expr	0,07	0,08	0,07	0,37	0,24
Gasch1	0,09	0,07	0,06	0,34	0,25
Gasch2	0,06	0,07	0,07	0,35	0,28
Pheno	0,06	0,09	0,08	0,33	0,32
Seq	0,07	0,07	0,07	0,42	0,28
Spo	0,08	0,07	0,07	0,36	0,31

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Serão apresentados os resultados obtidos com 50, 500 e 1000 épocas. As bases de dados selecionadas também foram a Church e Derisi para seguir o padrão já estabelecido. A Tabela 50 apresenta os resultados obtidos nas duas bases de dados com a execução de 50 épocas. Observa-se que em todos os limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. O mesmo ocorre na execução com 500 épocas conforme mostra a Tabela 51 e na execução com 1000 épocas como mostra a Tabela 52.

Comparando os resultados obtidos e mostrados nas três tabelas anteriores a Figura 72 apresenta os resultados da base de dados Church e a os resultados obtidos da base de dados Derisi são mostrados na Figura 73.

Analisando a medida de revocação, em todas bases de dados da ontologia Função Celular e com todos os limiares, o algoritmo MHC-CNN comparado como o Clus-HMC foi superior em mais de 97% dos casos conforme mostra a Figura 74. Nota-se, que com o aumento do número de épocas o percentual da diferença entre os algoritmos diminuiu.

A mesma comparação é feita entre o algoritmo MHC-CNN e o Clus-HSC, conforme mostra a Figura 75.

Tabela 50 – Comparativo da Medida de Precisão e Revocação com 50 épocas na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	**1,00**	0,00	1,00	0,00	1,00	0,00	**1,00**	0,00	1,00	0,00	1,00
T(10)	0,00	**0,92**	0,31	0,40	0,25	0,40	0,01	**0,89**	0,31	0,37	0,20	0,38
T(20)	0,01	**0,88**	0,54	0,30	0,44	0,31	0,01	**0,87**	0,62	0,26	0,36	0,31
T(30)	0,01	**0,85**	0,68	0,26	0,60	0,27	0,01	**0,86**	0,73	0,24	0,53	0,25
T(40)	0,01	**0,84**	0,91	0,20	0,80	0,21	0,01	**0,84**	0,83	0,20	0,67	0,22
T(50)	0,01	**0,81**	0,93	0,20	0,85	0,21	0,01	**0,83**	0,83	0,20	0,84	0,17
T(60)	0,02	**0,70**	0,94	0,20	0,89	0,21	0,01	**0,82**	0,99	0,17	0,89	0,17
T(70)	0,05	**0,50**	0,95	0,20	0,93	0,20	0,01	**0,81**	0,99	0,17	0,94	0,17
T(80)	0,14	**0,29**	0,99	0,18	0,99	0,17	0,01	**0,74**	0,99	0,17	0,96	0,17
T(90)	0,25	**0,25**	0,99	0,17	0,99	0,17	0,04	**0,48**	1,00	0,16	0,98	0,16
T(100)	0,26	**0,25**	1,00	0,17	0,99	0,17	0,21	**0,27**	1,00	0,16	0,98	0,16

Tabela 51 – Comparativo da Medida de Precisão e Revocação com 500 épocas na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	**1,00**	0,00	1,00	0,00	1,00	0,00	**1,00**	0,00	1,00	0,00	1,00
T(10)	0,00	**0,91**	0,31	0,40	0,25	0,40	0,01	**0,85**	0,31	0,37	0,20	0,38
T(20)	0,01	**0,87**	0,54	0,30	0,44	0,31	0,01	**0,82**	0,62	0,26	0,36	0,31
T(30)	0,01	**0,85**	0,68	0,26	0,60	0,27	0,01	**0,81**	0,73	0,24	0,53	0,25
T(40)	0,01	**0,83**	0,91	0,20	0,80	0,21	0,01	**0,81**	0,83	0,20	0,67	0,22
T(50)	0,01	**0,79**	0,93	0,20	0,85	0,21	0,01	**0,80**	0,83	0,20	0,84	0,17
T(60)	0,02	**0,69**	0,94	0,20	0,89	0,21	0,02	**0,65**	0,99	0,17	0,89	0,17
T(70)	0,03	**0,57**	0,95	0,20	0,93	0,20	0,09	**0,35**	0,99	0,17	0,94	0,17
T(80)	0,04	**0,53**	0,99	0,18	0,99	0,17	0,11	**0,32**	0,99	0,17	0,96	0,17
T(90)	0,07	**0,42**	0,99	0,17	0,99	0,17	0,16	**0,26**	1,00	0,16	0,98	0,16
T(100)	0,26	**0,26**	1,00	0,17	0,99	0,17	0,29	**0,24**	1,00	0,16	0,98	0,16

Aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 50 épocas e o Clus-HMC tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 500 épocas e 1000 épocas.

Tabela 52 – Comparativo da Medida de Precisão e Revocação com 1000 épocas na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	1,00	0,00	1,00	0,00	1,00	0,00	1,00	0,00	1,00	0,00	1,00
T(10)	0,00	0,90	0,31	0,40	0,25	0,40	0,01	0,85	0,31	0,37	0,20	0,38
T(20)	0,01	0,86	0,54	0,30	0,44	0,31	0,01	0,82	0,62	0,26	0,36	0,31
T(30)	0,01	0,84	0,68	0,26	0,60	0,27	0,01	0,81	0,73	0,24	0,53	0,25
T(40)	0,01	0,82	0,91	0,20	0,80	0,21	0,01	0,81	0,83	0,20	0,67	0,22
T(50)	0,01	0,79	0,93	0,20	0,85	0,21	0,01	0,79	0,83	0,20	0,84	0,17
T(60)	0,01	0,69	0,94	0,20	0,89	0,21	0,03	0,56	0,99	0,17	0,89	0,17
T(70)	0,02	0,60	0,95	0,20	0,93	0,20	0,11	0,32	0,99	0,17	0,94	0,17
T(80)	0,03	0,55	0,99	0,18	0,99	0,17	0,11	0,32	0,99	0,17	0,96	0,17
T(90)	0,05	0,48	0,99	0,17	0,99	0,17	0,11	0,32	1,00	0,16	0,98	0,16
T(100)	0,26	0,25	1,00	0,17	0,99	0,17	0,29	0,24	1,00	0,16	0,98	0,16

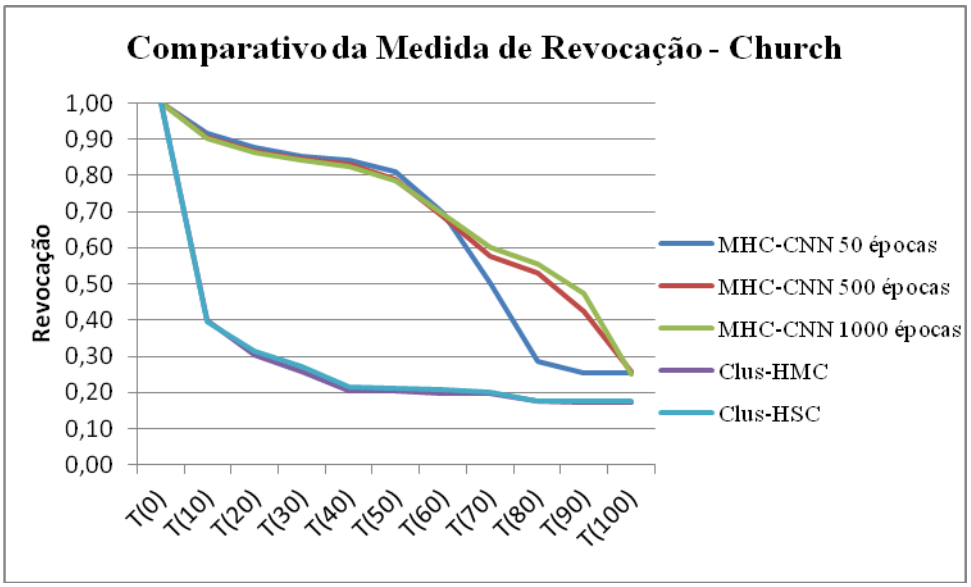


Figura 72 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Função Molecular.

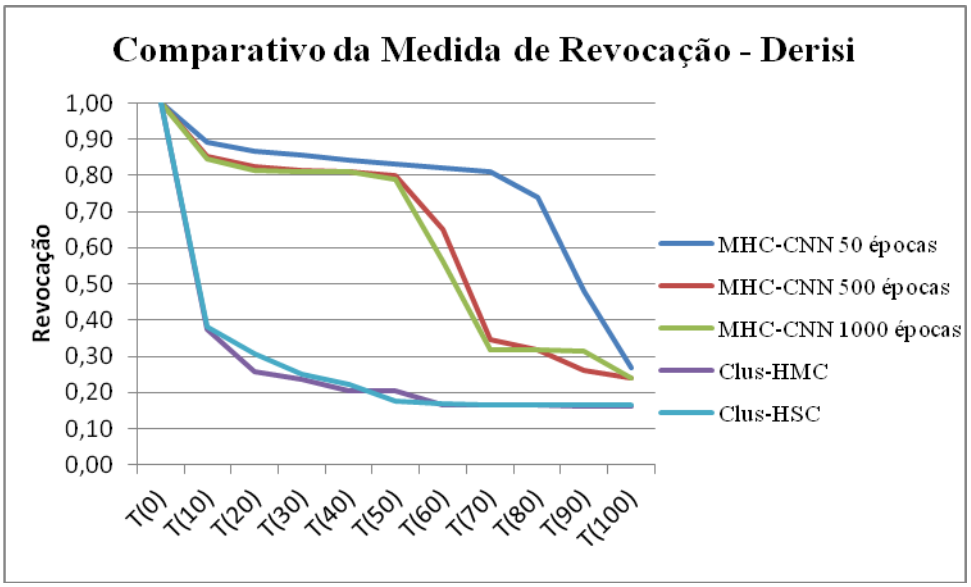


Figura 73 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Função Molecular.

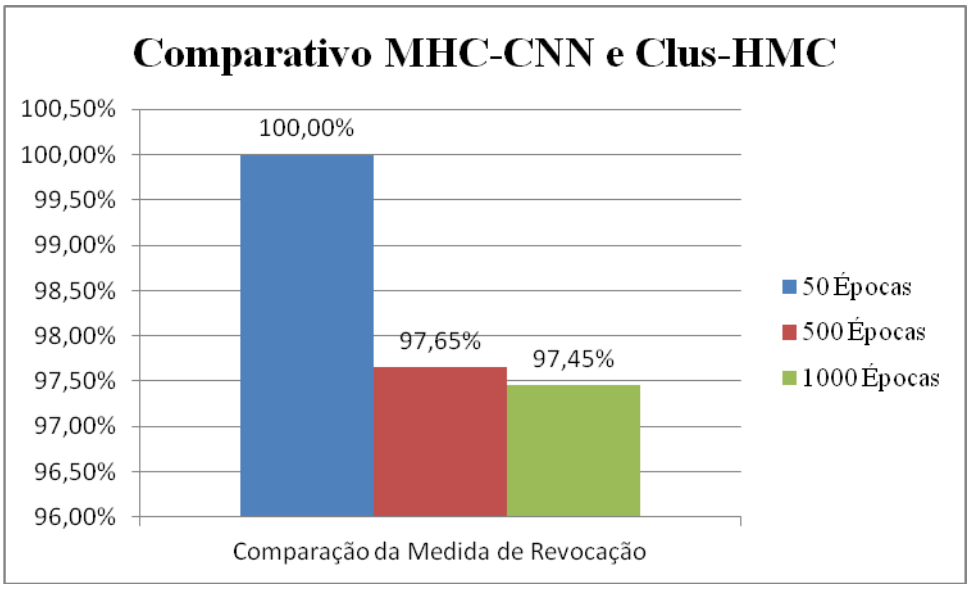


Figura 74 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Ontologia Função Molecular.

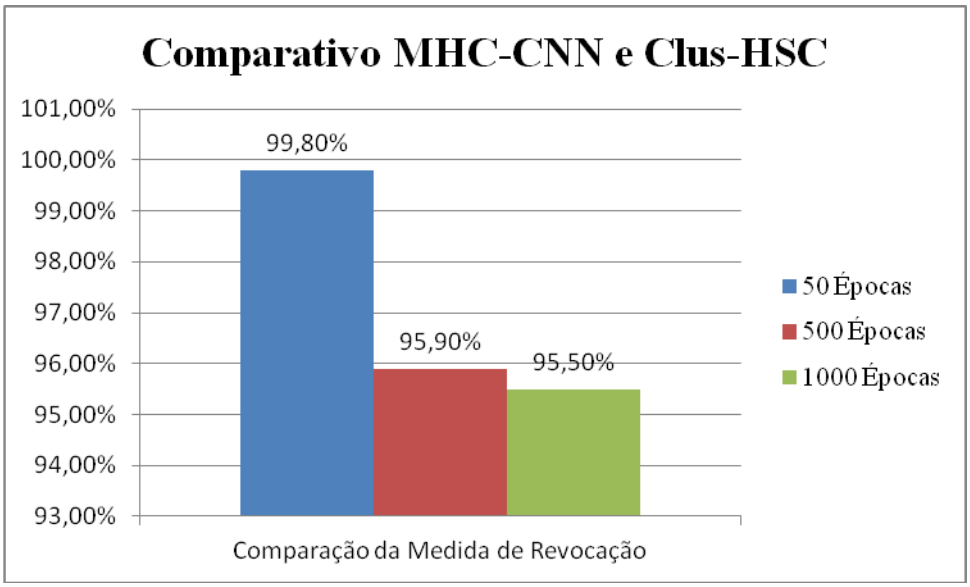


Figura 75 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Ontologia Função Molecular.

B1.3 Comparação MHC-ES e Clus

Tabela 53 mostra os resultados dos algoritmos utilizando a medida AUPRC com 20, 60 e 100 gerações. Aplicando o teste de Friedman têm-se $X_F^2 = 32{,}5$ e $F_F = 39{,}3$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Tabela 53 – Comparativo da Medida AUPRC entre o MHC-ES e o Clus-HMC e Clus-HSC na Ontologia Função Molecular.

	MHC-ES (em gerações)			Clus	
	20	**60**	**100**	**HMC**	**HSC**
Cellcycle	0,08	0,07	0,08	0,34	0,27
Church	0,08	0,05	0,05	0,36	0,34
Derisi	0,10	0,06	0,07	0,34	0,29
Eisen	0,05	0,07	0,06	0,34	0,28
Expr	0,05	0,05	0,05	0,37	0,24
Gasch1	0,09	0,06	0,06	0,34	0,25
Gasch2	0,07	0,06	0,06	0,35	0,28
Pheno	0,05	0,07	0,07	0,33	0,32
Seq	0,07	0,07	0,06	0,42	0,28
Spo	0,07	0,07	0,07	0,36	0,31

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC foi estatisticamente superior ao MHC-ES nas três gerações (20, 60 e 100). Já o Clus-HSC foi estatisticamente superior ao MHC-ES na execução com 60 e 100 gerações.

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Serão apresentados os resultados obtidos com 20, 60 e 100 gerações. As bases de dados selecionadas também foram a Church e Derisi seguindo a estratégia estabelecida. A Tabela 54 apresenta os resultados obtidos nas duas bases de dados com a execução de 20 gerações. Observa-se que em todos os limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. Já na execução do algoritmo MHC-ES com 60 e 100 gerações, conforme mostra a Tabela 55 e a Tabela 56 respectivamente, ocorreu empate nos dois últimos limiares comparados ao algoritmo Clus-HMC.

Tabela 54 – Comparativo da Medida de Precisão e Revocação com 20 gerações na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clu s-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	**1,00**	0,00	1,00	0,00	1,00	0,00	**1,00**	0,00	1,00	0,00	1,00
T(10)	0,00	**1,00**	0,31	0,40	0,25	0,40	0,01	**0,89**	0,31	0,37	0,20	0,38
T(20)	0,00	**0,98**	0,54	0,30	0,44	0,31	0,01	**0,83**	0,62	0,26	0,36	0,31
T(30)	0,00	**0,94**	0,68	0,26	0,60	0,27	0,01	**0,80**	0,73	0,24	0,53	0,25
T(40)	0,01	**0,87**	0,91	0,20	0,80	0,21	0,01	**0,76**	0,83	0,20	0,67	0,22
T(50)	0,01	**0,73**	0,93	0,20	0,85	0,21	0,02	**0,64**	0,83	0,20	0,84	0,17
T(60)	0,01	**0,46**	0,94	0,20	0,89	0,21	0,03	**0,58**	0,99	0,17	0,89	0,17
T(70)	0,03	**0,30**	0,95	0,20	0,93	0,20	0,05	**0,46**	0,99	0,17	0,94	0,17
T(80)	0,07	**0,19**	0,99	0,18	0,99	0,17	0,09	**0,27**	0,99	0,17	0,96	0,17
T(90)	0,18	**0,18**	0,99	0,17	0,99	0,17	0,15	**0,18**	1,00	0,16	0,98	0,16
T(100)	0,36	**0,18**	1,00	0,17	0,99	0,17	0,37	**0,18**	1,00	0,16	0,98	0,16

Comparando os resultados obtidos e mostrados nas três tabelas apresentadas anteriormente a Figura 76 apresenta os resultados da base de dados Church e a os resultados obtidos da base de dados Derisi são mostrados na Figura 77.

Tabela 55 – Comparativo da Medida de Precisão e Revocação com 60 gerações na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	**1,00**	0,00	1,00	0,00	1,00	0,00	**1,00**	0,00	1,00	0,00	1,00
T(10)	0,00	**1,00**	0,31	0,40	0,25	0,40	0,01	**0,84**	0,31	0,37	0,20	0,38
T(20)	0,00	**0,98**	0,54	0,30	0,44	0,31	0,01	**0,79**	0,62	0,26	0,36	0,31
T(30)	0,00	**0,91**	0,68	0,26	0,60	0,27	0,01	**0,76**	0,73	0,24	0,53	0,25
T(40)	0,01	**0,77**	0,91	0,20	0,80	0,21	0,02	**0,66**	0,83	0,20	0,67	0,22
T(50)	0,01	**0,53**	0,93	0,20	0,85	0,21	0,03	**0,47**	0,83	0,20	0,84	0,17
T(60)	0,03	**0,39**	0,94	0,20	0,89	0,21	0,05	**0,41**	0,99	0,17	0,89	0,17
T(70)	0,08	**0,19**	0,95	0,20	0,93	0,20	0,08	**0,26**	0,99	0,17	0,94	0,17
T(80)	0,12	**0,17**	0,99	0,18	0,99	0,17	0,09	**0,17**	0,99	0,17	0,96	0,17
T(90)	0,20	**0,17**	0,99	0,17	0,99	0,17	0,09	**0,17**	1,00	0,16	0,98	0,16
T(100)	0,20	**0,17**	1,00	0,17	0,99	0,17	0,21	**0,17**	1,00	0,16	0,98	0,16

Tabela 56 – Comparativo da Medida de Precisão e Revocação com 100 gerações na Ontologia Função Molecular.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,00	**1,00**	0,00	1,00	0,00	1,00	0,00	**1,00**	0,00	1,00	0,00	1,00
T(10)	0,00	**1,00**	0,31	0,40	0,25	0,40	0,01	**0,84**	0,31	0,37	0,20	0,38
T(20)	0,00	**0,97**	0,54	0,30	0,44	0,31	0,01	**0,76**	0,62	0,26	0,36	0,31
T(30)	0,00	**0,91**	0,68	0,26	0,60	0,27	0,01	**0,73**	0,73	0,24	0,53	0,25
T(40)	0,01	**0,80**	0,91	0,20	0,80	0,21	0,02	**0,65**	0,83	0,20	0,67	0,22
T(50)	0,01	**0,56**	0,93	0,20	0,85	0,21	0,03	**0,54**	0,83	0,20	0,84	0,17
T(60)	0,02	**0,33**	0,94	0,20	0,89	0,21	0,07	**0,36**	0,99	0,17	0,89	0,17
T(70)	0,12	**0,17**	0,95	0,20	0,93	0,20	0,11	**0,27**	0,99	0,17	0,94	0,17
T(80)	0,17	**0,17**	0,99	0,18	0,99	0,17	0,15	**0,17**	0,99	0,17	0,96	0,17
T(90)	0,18	**0,17**	0,99	0,17	0,99	0,17	0,16	**0,17**	1,00	0,16	0,98	0,16
T(100)	0,20	**0,17**	1,00	0,17	0,99	0,17	0,21	**0,17**	1,00	0,16	0,98	0,16

Analisando a medida de revocação, em todas bases de dados da ontologia Função Celular e com todos os limiares, o algoritmo MHC-ES comparado como o Clus-HMC foi superior em mais de 86% dos casos conforme mostra a Figura 78. A mesma comparação é feita entre o algoritmo MHC-ES e o Clus-HSC, conforme mostra a Figura 79 em que mais de 95% dos casos a medida de revocação foi superior no algoritmo MHC-ES.

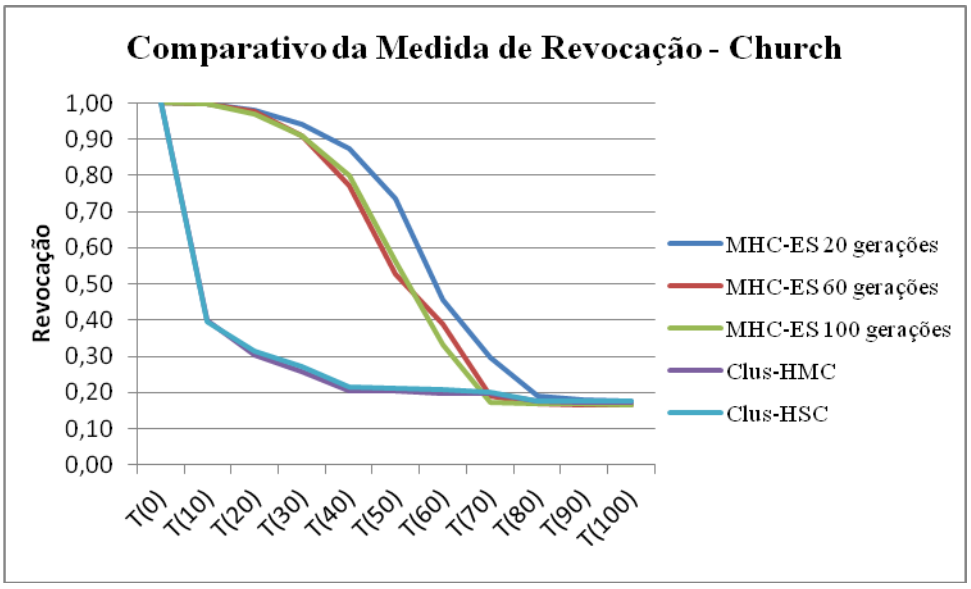


Figura 76 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Função Molecular.

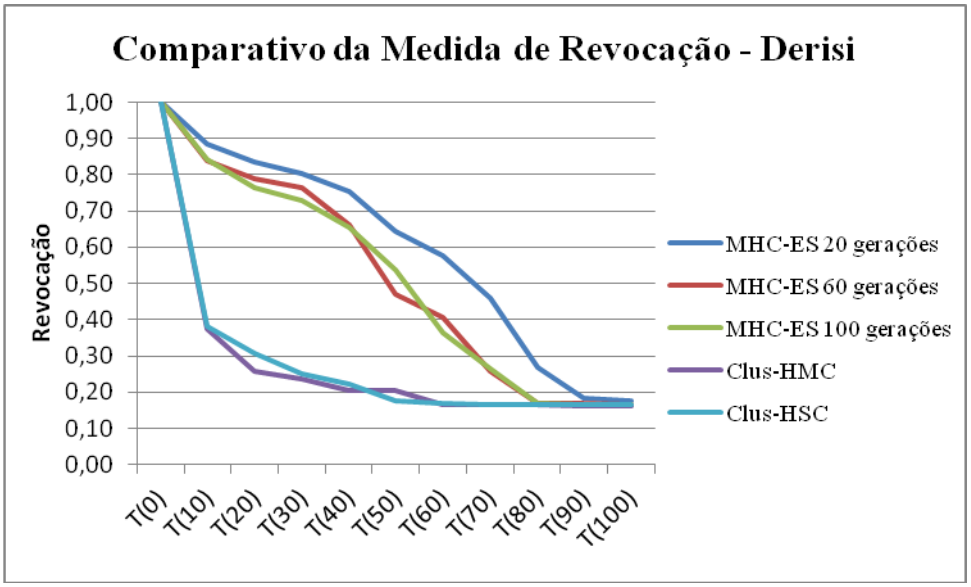


Figura 77 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Função Molecular.

Aplicando o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 20 gerações e o Clus-HMC tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 60 épocas e 100 gerações.

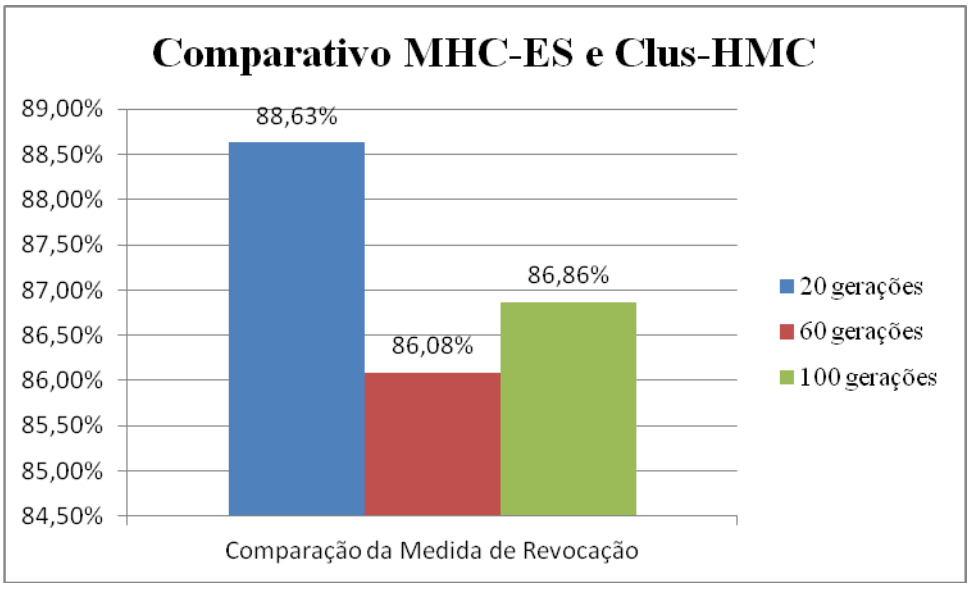


Figura 78 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Ontologia Função Molecular.

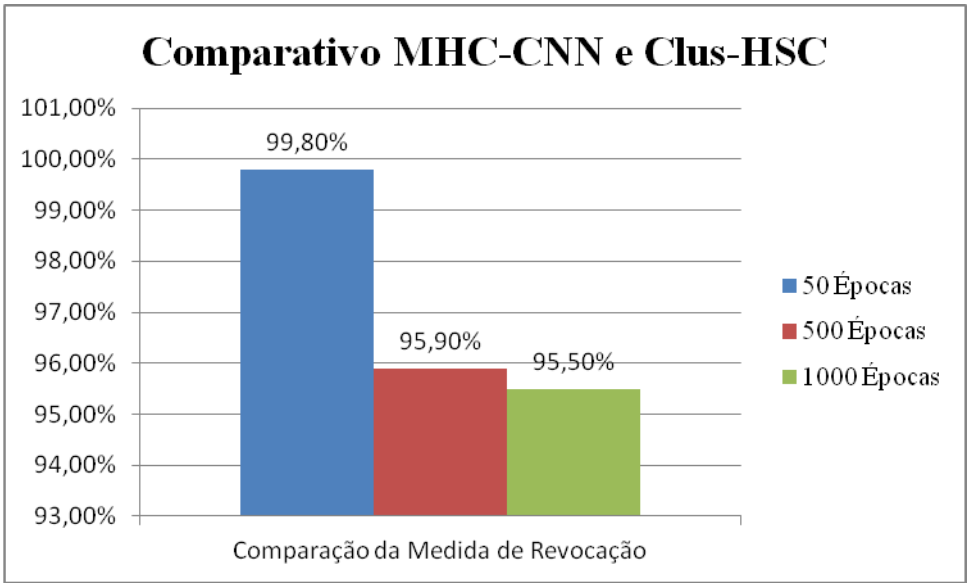


Figura 79 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Ontologia Função Molecular.

B2 ONTOLOGIA PROCESSO BIOLÓGICO

Nesta Seção são apresentados os comparativos realizados, na Ontologia Processo Biológico, entre os algoritmos: MHC-CNN x MHC-ES, MHC-CNN x Clus (HMC e HSC) e MHC-ES x Clus (HMC e HSC).

B2.1 Comparação MHC-CNN e MHC-ES

A Tabela 57 mostra os resultados dos algoritmos MHC-CNN e MHC-ES na base de dados da ontologia Processo Biológico, quando utilizado a medida de distância.

Tabela 57 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida de distância na Ontologia Processo Biológico.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	57,1%	57,3%	57,4%	39,5%	41,3%	41,3%
Church	54,7%	55,3%	55,3%	60,5%	61,5%	60,3%
Derisi	52,2%	52,3%	52,7%	55,4%	57,4%	55,7%
Eisen	57,5%	56,0%	56,4%	47,4%	41,7%	48,1%
Expr	60,5%	59,8%	60,7%	46,8%	49,2%	48,0%
Gasch1	59,4%	58,9%	58,4%	39,1%	35,1%	38,3%
Gasch2	57,2%	57,1%	57,5%	39,3%	45,2%	59,3%
Pheno	48,6%	47,2%	47,2%	61,0%	58,9%	55,7%
Seq	59,2%	59,5%	61,3%	56,5%	55,7%	55,7%
Spo	55,1%	55,3%	55,0%	61,1%	61,3%	58,6%

Calculando os valores para as equações utilizadas no teste de Friedman têm-se $X_F^2 = 1{,}26$ e $F_F = 0{,}2$. Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2{,}09$. Deste modo a hipótese nula não é rejeitada, $F_F < F(9;45)$ indicando que não há diferença significativa entre os resultados dos algoritmos.

A Tabela 58 mostra os resultados dos dois algoritmos, quando utilizado a medida hF. Calculando os valores para as equações utilizadas no teste de Friedman tem-se $X_F^2 = 35{,}9$ e $F_F = 22{,}9$.

Conforme o valor tabelado para os graus de liberdade 9 e 45 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;45) = 2{,}09$. Deste modo a $F_F > F(9;45)$ logo a hipótese nula não pode ser rejeitada, indicando que há diferença estatística entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 2{,}38$. Considerando os dados na Tabela 58 , o algoritmo MHC-CNN nas três épocas selecionadas (50, 500 e 1000), é estatisticamente superior ao algoritmo MHC-ES com 20 gerações. Além disso, o MHC-CNN, com 50 e 500 épocas, é estatisticamente superior ao resultado do algoritmo MHC-ES com 60 e 100 gerações.

Tabela 58 – Comparativo entre o MHC-CNN e o MHC-ES usando a medida-Fh na Ontologia Processo Biológico.

	MHC-CNN (em épocas)			MHC-ES (em gerações)		
	50	**500**	**1000**	**20**	**60**	**100**
Cellcycle	36,3%	37,7%	36,9%	18,2%	23,0%	22,1%
Church	30,1%	30,9%	30,5%	28,8%	28,7%	30,1%
Derisi	32,7%	30,5%	30,9%	28,9%	32,4%	29,1%
Eisen	32,7%	37,9%	37,7%	25,8%	22,3%	25,6%
Expr	42,5%	42,2%	42,1%	22,2%	26,8%	27,4%
Gasch1	38,9%	38,3%	37,9%	20,1%	15,9%	18,5%
Gasch2	35,8%	35,4%	35,2%	16,0%	25,3%	32,9%
Pheno	35,2%	34,6%	34,4%	29,7%	31,2%	29,1%
Seq	43,5%	43,3%	44,0%	28,2%	29,0%	29,3%
Spo	33,5%	32,9%	32,1%	28,7%	31,6%	28,2%

B1.2 Comparação MHC-CNN e Clus

A Tabela 59 mostra os resultados dos algoritmos utilizando a medida AUPRC com 50, 500 e 1000 épocas.

Tabela 59 – Comparativo da Medida AUPRC entre o MHC-CNN e o Clus-HMC e Clus-HSC na Ontologia Processo Biológico.

	MHC-CNN (em épocas)			Clus	
	50	**500**	**1000**	**HMC**	**HSC**
Cellcycle	0,12	0,13	0,13	0,35	0,24
Church	0,12	0,12	0,12	0,36	0,29
Derisi	0,10	0,13	0,12	0,33	0,27
Eisen	0,12	0,11	0,10	0,41	0,26
Expr	0,09	0,14	0,15	0,39	0,21
Gasch1	0,15	0,15	0,14	0,37	0,22
Gasch2	0,13	0,08	0,09	0,35	0,25
Pheno	0,08	0,11	0,12	0,32	0,30
Seq	0,08	0,08	0,08	0,37	0,20
Spo	0,09	0,13	0,11	0,35	0,24

Calculando os valores têm-se $X_F^2 = 32{,}4$ e $F_F = 38{,}5$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo

Clus-HMC foi estatisticamente superior ao MHC-CNN nas três épocas (50, 500 e 1000). Já o Clus-HSC foi estatisticamente superior ao MHC-CNN na execução com 50 e 1000 épocas.

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Serão apresentados os resultados obtidos com 50, 500 e 1000 épocas. As bases de dados selecionadas também foram a Church e Derisi. A Tabela 60 apresenta os resultados obtidos nas duas bases de dados com a execução de 50 épocas. Observa-se que na base Church apenas o último limiar foi inferior ao resultado do algoritmo MHC-CNN. Já na base Derisi todos os limiares o resultado da medida de revocação foi superior comparado com os dois outros algoritmos.

Tabela 60 – Comparativo da Medida de Precisão e Revocação com 50 épocas na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clu s-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0,0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10,0)	0,01	**0,91**	0,26	0,48	0,30	0,37	0,01	**0,87**	0,26	0,45	0,26	0,37
T(20,0)	0,01	**0,87**	0,45	0,35	0,41	0,31	0,01	**0,85**	0,43	0,32	0,38	0,29
T(30,0)	0,01	**0,84**	0,58	0,28	0,53	0,24	0,01	**0,84**	0,59	0,25	0,47	0,23
T(40,0)	0,01	**0,82**	0,66	0,24	0,60	0,21	0,01	**0,82**	0,67	0,22	0,54	0,20
T(50,0)	0,02	**0,80**	0,72	0,20	0,69	0,16	0,02	**0,81**	0,71	0,19	0,61	0,18
T(60,0)	0,02	**0,75**	0,84	0,14	0,76	0,13	0,02	**0,79**	0,80	0,14	0,71	0,14
T(70,0)	0,05	**0,62**	0,86	0,13	0,80	0,12	0,02	**0,74**	0,89	0,10	0,79	0,11
T(80,0)	0,14	**0,37**	0,92	0,08	0,84	0,09	0,02	**0,70**	0,89	0,10	0,86	0,08
T(90,0)	0,23	**0,25**	0,95	0,07	0,92	0,07	0,04	**0,57**	0,99	0,06	0,93	0,07
T(100,0)	0,34	0,06	0,97	**0,07**	0,94	0,06	0,25	**0,11**	1,00	0,06	0,97	0,06

Já com a execução com 500 e 1000 épocas em todos os limiares a medida de revocação do algoritmo MHC-CNN foi superior ao Clus-HMC e Clus-HSC, conforme mostra Tabela 61 a e a Tabela 62, respectivamente.

Tabela 61 – Comparativo da Medida de Precisão e Revocação com 500 épocas na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0,0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10,0)	0,01	**0,89**	0,26	0,48	0,30	0,37	0,02	**0,80**	0,26	0,45	0,26	0,37
T(20,0)	0,01	**0,84**	0,45	0,35	0,41	0,31	0,02	**0,76**	0,43	0,32	0,38	0,29
T(30,0)	0,01	**0,81**	0,58	0,28	0,53	0,24	0,02	**0,73**	0,59	0,25	0,47	0,23
T(40,0)	0,02	**0,79**	0,66	0,24	0,60	0,21	0,02	**0,71**	0,67	0,22	0,54	0,20
T(50,0)	0,02	**0,76**	0,72	0,20	0,69	0,16	0,02	**0,70**	0,71	0,19	0,61	0,18
T(60,0)	0,03	**0,71**	0,84	0,14	0,76	0,13	0,04	**0,61**	0,80	0,14	0,71	0,14
T(70,0)	0,05	**0,62**	0,86	0,13	0,80	0,12	0,12	**0,36**	0,89	0,10	0,79	0,11
T(80,0)	0,08	**0,50**	0,92	0,08	0,84	0,09	0,12	**0,36**	0,89	0,10	0,86	0,08
T(90,0)	0,13	**0,39**	0,95	0,07	0,92	0,07	0,16	**0,32**	0,99	0,06	0,93	0,07
T(100,0)	0,33	**0,07**	0,97	0,07	0,94	0,06	0,33	**0,24**	1,00	0,06	0,97	0,06

Tabela 62 – Comparativo da Medida de Precisão e Revocação com 1000 épocas na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-CNN		Clus-HMC		Clus-HSC		MHC-CNN		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0,0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10,0)	0,01	**0,88**	0,26	0,48	0,30	0,37	0,02	**0,79**	0,26	0,45	0,26	0,37
T(20,0)	0,01	**0,84**	0,45	0,35	0,41	0,31	0,02	**0,74**	0,43	0,32	0,38	0,29
T(30,0)	0,02	**0,81**	0,58	0,28	0,53	0,24	0,02	**0,72**	0,59	0,25	0,47	0,23
T(40,0)	0,02	**0,79**	0,66	0,24	0,60	0,21	0,02	**0,71**	0,67	0,22	0,54	0,20
T(50,0)	0,02	**0,76**	0,72	0,20	0,69	0,16	0,03	**0,69**	0,71	0,19	0,61	0,18
T(60,0)	0,03	**0,71**	0,84	0,14	0,76	0,13	0,05	**0,56**	0,80	0,14	0,71	0,14
T(70,0)	0,04	**0,64**	0,86	0,13	0,80	0,12	0,10	**0,40**	0,89	0,10	0,79	0,11
T(80,0)	0,07	**0,56**	0,92	0,08	0,84	0,09	0,11	**0,36**	0,89	0,10	0,86	0,08
T(90,0)	0,11	**0,41**	0,95	0,07	0,92	0,07	0,13	**0,35**	0,99	0,06	0,93	0,07
T(100,0)	0,27	**0,10**	0,97	0,07	0,94	0,06	0,30	**0,24**	1,00	0,06	0,97	0,06

Comparando os resultados obtidos e mostrados nas três tabelas anteriores a Figura 80 apresenta os resultados da base de dados Church e a os resultados obtidos da base de dados Derisi são mostrados na Figura 81.

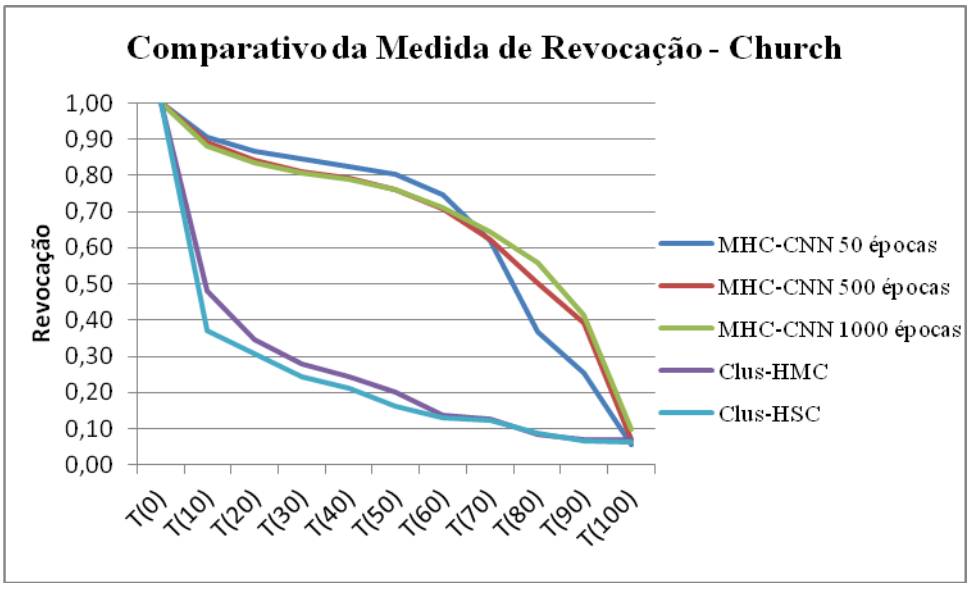


Figura 80 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Processo Biológico.

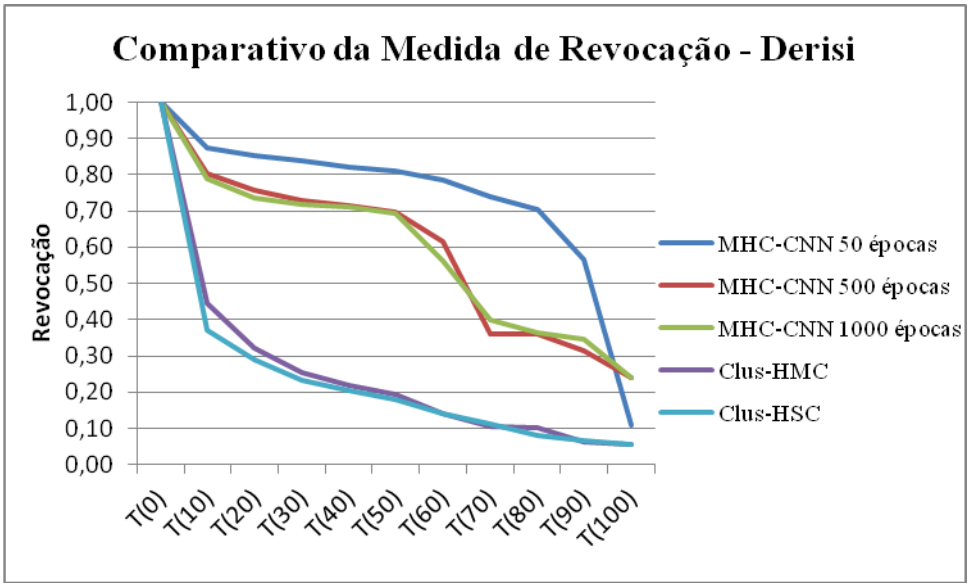


Figura 81 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Processo Biológico.

Analisando a medida de revocação, em todas bases de dados da ontologia Processo Biológico e com todos os limiares, o algoritmo MHC-CNN comparado como o Clus-HMC foi superior em mais de 99% dos casos conforme mostra a Figura 82, chegando a 100% na execução do algoritmo com 500 épocas.

A mesma comparação é feita entre o algoritmo MHC-CNN e o Clus-HSC, conforme mostra a Figura 83, em que mais de 97% dos casos a medida de revocação foi superior no algoritmo MHC-CNN.

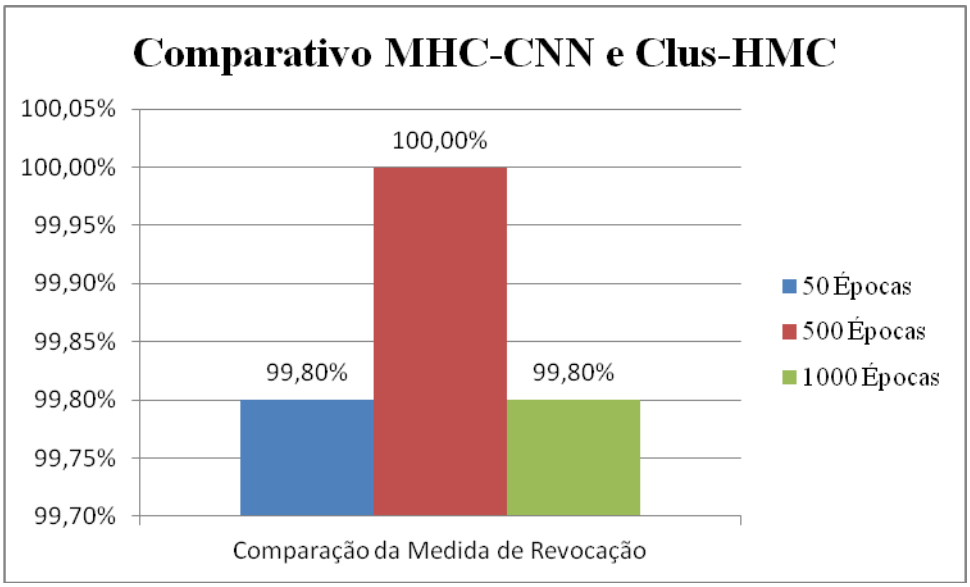


Figura 82 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Ontologia Processo Biológico.

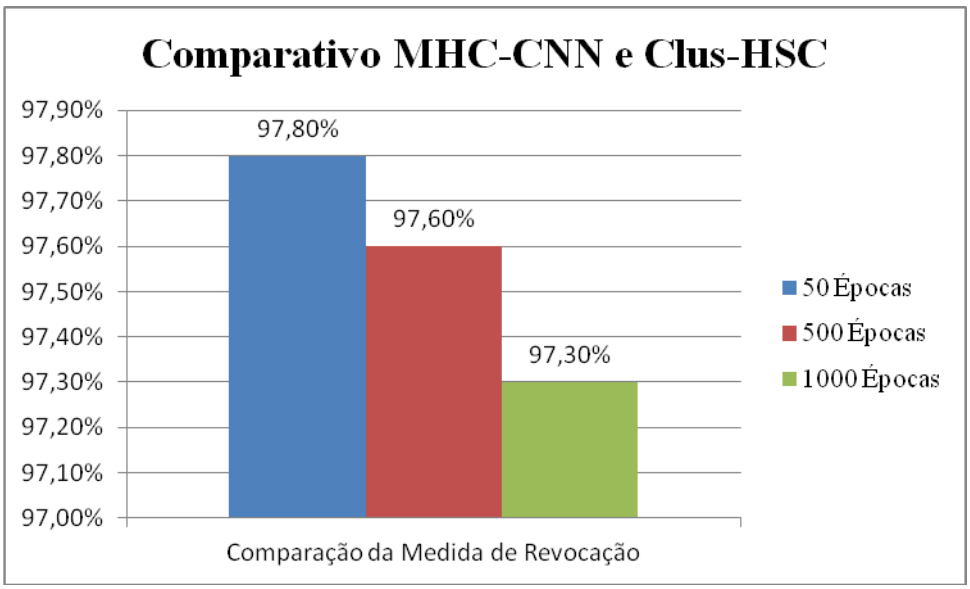


Figura 83 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Ontologia Processo Biológico.

Aplicado o teste de Wilcoxon tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-CNN com 500 épocas e 1000.

B1.3 Comparação MHC-ES e Clus

A Tabela 63 mostra os resultados dos algoritmos utilizando a medida AUPRC com 20, 60 e 100 gerações. Aplicando o teste de Friedman têm-se $X_F^2 = 32{,}0$ e $F_F = 36{,}1$. De acordo com o valor tabelado para os graus de liberdade 9 e 36 na distribuição F de Snedecor encontra-se o valor crítico de $F(9;36) = 2{,}15$. Desse modo a hipótese nula é rejeitada, pois $F_F > F(9;36)$ indicando que há diferença significativa entre os resultados dos algoritmos.

Tabela 63 – Comparativo da Medida AUPRC entre o MHC-ES e o Clus-HMC e Clus-HSC na Ontologia Processo Biológico.

	MHC-ES (em gerações)			Clus	
	20	**60**	**100**	**HMC**	**HSC**
Cellcycle	0,08	0,12	0,13	0,35	0,24
Church	0,03	0,03	0,05	0,36	0,29
Derisi	0,09	0,06	0,07	0,33	0,27
Eisen	0,08	0,10	0,09	0,41	0,26
Expr	0,03	0,03	0,03	0,39	0,21
Gasch1	0,06	0,05	0,05	0,37	0,22
Gasch2	0,07	0,04	0,05	0,35	0,25
Pheno	0,08	0,09	0,08	0,32	0,30
Seq	0,09	0,07	0,08	0,37	0,20
Spo	0,07	0,08	0,07	0,35	0,24

Considerando o nível de significância de 95% tem-se o valor de $CD = 1{,}93$. Através do valor obtido pela CD e verificando o resultado dos algoritmos observa-se que o algoritmo Clus-HMC e Clus-HSC foi estatisticamente superior ao MHC-ES nas três gerações (20, 60 e 100).

Também, avaliaram-se as duas medidas que dão origem a medida AUPRC: precisão e revocação nas 10 bases de dados.

Serão apresentados os resultados obtidos com 20, 60 e 100 gerações. As mesmas bases de dados foram selecionadas para realizar essa comparação: Church e Derisi. A Tabela 64 apresenta os resultados obtidos nas duas bases de dados com a execução de 20 gerações. Observa-se que na maioria dos limiares o resultado da medida de revocação foi superior, identificados em negrito, comparado com os dois outros algoritmos. O mesmo ocorre na execução do algoritmo MHC-ES com 60 e 100 gerações, conforme mostra a Tabela 65 e a Tabela 66, respectivamente.

Tabela 64 – Comparativo da Medida de Precisão e Revocação com 20 gerações na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clu s-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,26	0,48	0,30	0,37	0,02	**0,84**	0,26	0,45	0,26	0,37
T(20)	0,01	**0,99**	0,45	0,35	0,41	0,31	0,02	**0,81**	0,43	0,32	0,38	0,29
T(30)	0,01	**0,94**	0,58	0,28	0,53	0,24	0,03	**0,73**	0,59	0,25	0,47	0,23
T(40)	0,01	**0,84**	0,66	0,24	0,60	0,21	0,06	**0,54**	0,67	0,22	0,54	0,20
T(50)	0,01	**0,60**	0,72	0,20	0,69	0,16	0,09	**0,38**	0,71	0,19	0,61	0,18
T(60)	0,01	**0,29**	0,84	0,14	0,76	0,13	0,16	**0,27**	0,80	0,14	0,71	0,14
T(70)	0,05	**0,15**	0,86	0,13	0,80	0,12	0,12	**0,07**	0,89	0,10	0,79	0,11
T(80)	0,12	0,07	0,92	**0,08**	0,84	0,09	0,19	0,07	0,89	**0,10**	0,86	0,08
T(90)	0,17	**0,07**	0,95	0,07	0,92	0,07	0,20	**0,07**	0,99	0,06	0,93	0,07
T(100)	0,22	**0,07**	0,97	0,07	0,94	0,06	0,24	**0,06**	1,00	0,06	0,97	0,06

Tabela 65 – Comparativo da Medida de Precisão e Revocação com 60 gerações na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,26	0,48	0,30	0,37	0,02	**0,85**	0,26	0,45	0,26	0,37
T(20)	0,01	**0,99**	0,45	0,35	0,41	0,31	0,02	**0,80**	0,43	0,32	0,38	0,29
T(30)	0,01	**0,95**	0,58	0,28	0,53	0,24	0,02	**0,77**	0,59	0,25	0,47	0,23
T(40)	0,01	**0,82**	0,66	0,24	0,60	0,21	0,04	**0,62**	0,67	0,22	0,54	0,20
T(50)	0,02	**0,58**	0,72	0,20	0,69	0,16	0,06	**0,40**	0,71	0,19	0,61	0,18
T(60)	0,02	**0,31**	0,84	0,14	0,76	0,13	0,10	**0,31**	0,80	0,14	0,71	0,14
T(70)	0,08	0,07	0,86	**0,13**	0,80	0,12	0,06	**0,10**	0,89	0,10	0,79	0,11
T(80)	0,11	0,07	0,92	0,08	0,84	**0,09**	0,17	0,07	0,89	**0,10**	0,86	0,08
T(90)	0,16	**0,07**	0,95	0,07	0,92	0,07	0,18	**0,07**	0,99	0,06	0,93	0,07
T(100)	0,20	**0,07**	0,97	0,07	0,94	0,06	0,20	**0,06**	1,00	0,06	0,97	0,06

Tabela 66 – Comparativo da Medida de Precisão e Revocação com 100 gerações na Ontologia Processo Biológico.

Limiares	Church						Derisi					
	MHC-ES		Clus-HMC		Clus-HSC		MHC-ES		Clus-HMC		Clus-HSC	
	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.	Prec.	Rev.
T(0)	0,01	**1,00**	0,01	1,00	0,01	1,00	0,01	**1,00**	0,01	1,00	0,01	1,00
T(10)	0,01	**1,00**	0,26	0,48	0,30	0,37	0,02	**0,87**	0,26	0,45	0,26	0,37
T(20)	0,01	**0,99**	0,45	0,35	0,41	0,31	0,02	**0,81**	0,43	0,32	0,38	0,29
T(30)	0,01	**0,95**	0,58	0,28	0,53	0,24	0,03	**0,77**	0,59	0,25	0,47	0,23
T(40)	0,01	**0,83**	0,66	0,24	0,60	0,21	0,04	**0,60**	0,67	0,22	0,54	0,20
T(50)	0,02	**0,60**	0,72	0,20	0,69	0,16	0,06	**0,47**	0,71	0,19	0,61	0,18
T(60)	0,03	**0,33**	0,84	0,14	0,76	0,13	0,09	**0,31**	0,80	0,14	0,71	0,14
T(70)	0,13	**0,15**	0,86	0,13	0,80	0,12	0,09	0,09	0,89	0,10	0,79	**0,11**
T(80)	0,13	0,07	0,92	0,08	0,84	0,09	0,12	0,07	0,89	**0,10**	0,86	0,08
T(90)	0,15	**0,07**	0,95	0,07	0,92	0,07	0,18	**0,07**	0,99	0,06	0,93	0,07
T(100)	0,20	**0,07**	0,97	0,07	0,94	0,06	0,20	**0,06**	1,00	0,06	0,97	0,06

Comparando os resultados obtidos e mostrados nas três tabelas anteriores (Tabela 64, Tabela 65 e Tabela 66) apresentadas anteriormente a Figura 84 apresenta os resultados da base de dados Church e a os resultados obtidos da base de dados Derisi são mostrados na Figura 85.

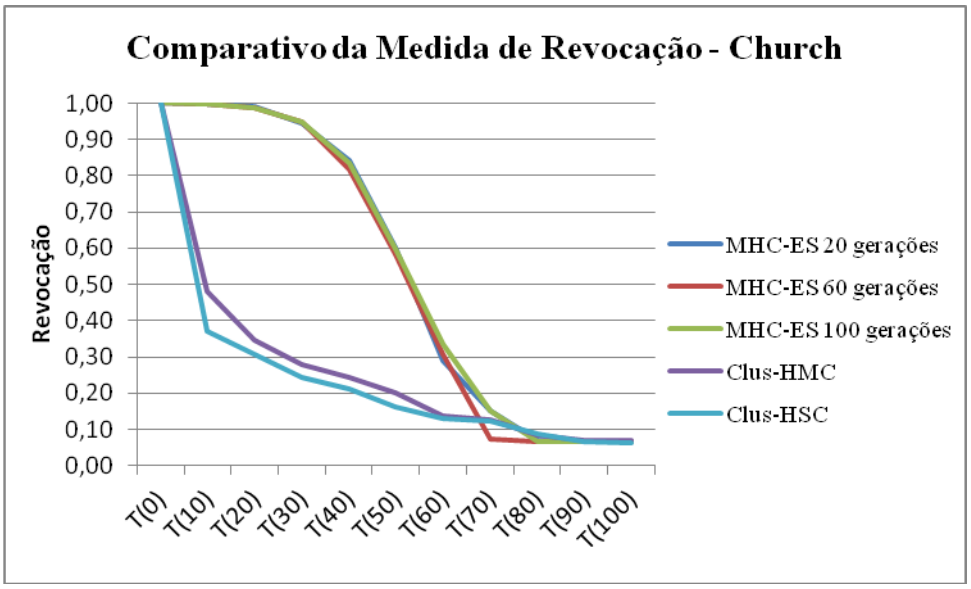


Figura 84 – Comparativo Geral da Medida de Revocação na Base de Dados Church da Ontologia Processo Biológico.

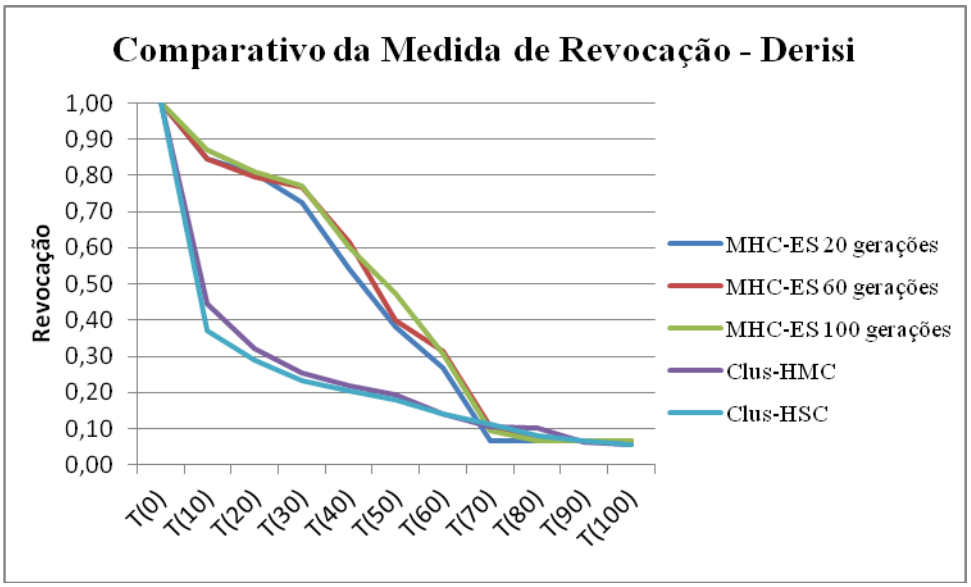


Figura 85 – Comparativo Geral da Medida de Revocação na Base de Dados Derisi da Ontologia Processo Biológico.

Analisando a medida de revocação, em todas bases de dados da ontologia Processo Biológico e com todos os limiares, o algoritmo MHC-ES comparado como o Clus-HMC foi superior em mais de 83% dos casos conforme mostra a Figura 86. A mesma comparação é feita entre o algoritmo MHC-ES e o Clus-HSC, conforme mostra a Figura 87 em mais de 82% dos casos.

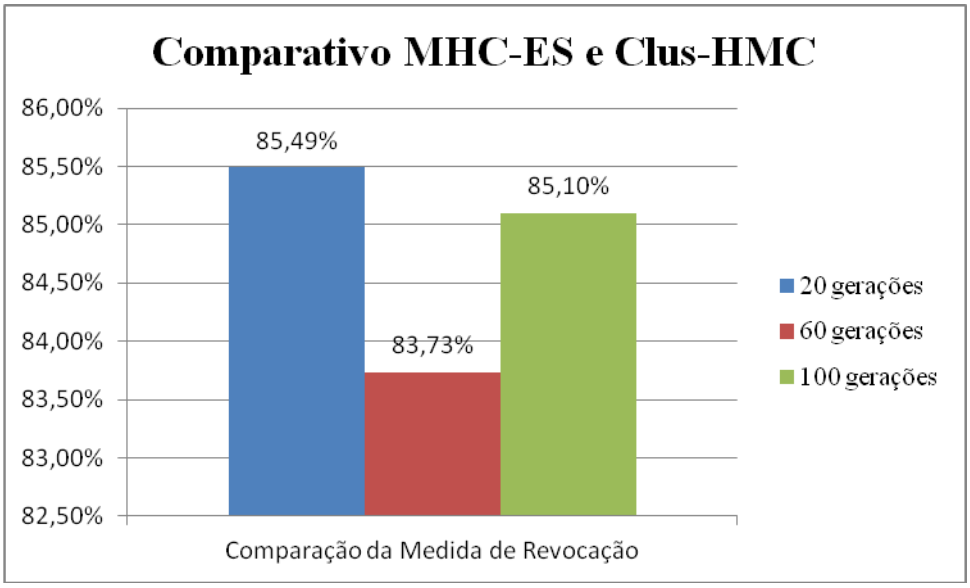


Figura 86 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HMC na Ontologia Processo Biológico.

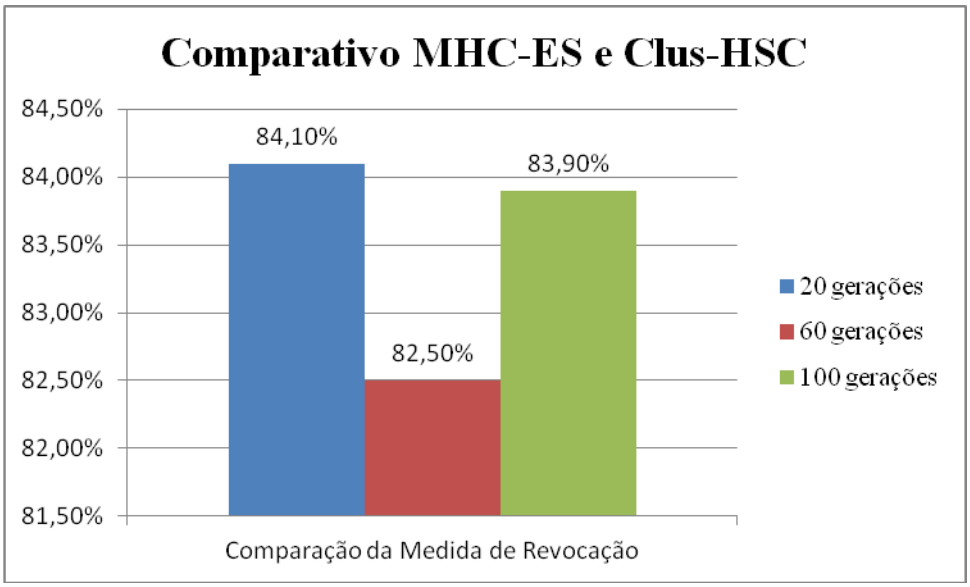


Figura 87 – Comparação Geral da Medida de Revocação entre o MHC-CNN e o Clus-HSC na Ontologia Processo Biológico.

Aplicando o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 20 gerações e o Clus-HMC tem-se $T = \min(0;55) = 0$. Considerando que o valor de $n(10 < 25)$ é tabelado para pequenas amostras, os valores críticos com nível de confiança $\alpha = 0{,}05$ a hipótese nula deve ser rejeitada assumindo que há diferença entre os algoritmos.

O mesmo resultado foi obtido quando aplicado o teste de Wilcoxon com os resultados obtidos pelo algoritmo MHC-ES com 60 gerações e 100 gerações.