GAESC-SG: Um algoritmo genético para evolução de classificadores simbólicos utilizando operadores de especialização e generalização de classificadores

Carlos David Ribeiro Pasco[1], **Flavia C. Bernardini**[1]

[1] Laboratório de Inovação no Desenvolvimento de Sistemas (LabIDeS)
Departamento de Computação – Instituto de Ciência e Tecnologia
Universidade Federal Fluminense (UFF) – Rio das Ostras, RJ – Brazil

cpasco@gmail.com, fcbernardini@puro.uff.br

Abstract. *In many real problems, it is important to evaluate the precision of classifiers, as well as analyze the knowledge of this classifier to discover new knowledge or to offer some explanation of the given classification. Symbolic classifiers are indicated when knowledge acquisition is needed, but, in some domains, they show precision rates lower than precision rates of classifiers induced by other types of learning algorithms. One solution is to evolve symbolic classifiers into only one symbolic classifier. In this work we propose GAESC-SG, a genetic algorithm that evolves symbolic classifiers into a single classifier. This algorithm uses crossover and mutation operators and also uses specialization and generalization operators. The experiments realized with this algorithm shows interesting results, because GAESC-SG shows more stable results on number of rules and generations, related to GAESC.*

Resumo. *Em muitos problemas do mundo real, além de ser importante avaliar a precisão de classificadores construídos, muitas vezes também é importante analisar o conhecimento nele contido, com o objetivo de descobrir novo conhecimento e/ou fornecer explicação da classificação dada. Classificadores simbólicos são indicados nesses casos, mas em alguns domínios apresentam taxas de acerto menores que classificadores induzidos por outros tipos de algoritmos de aprendizado. Uma solução é evoluir classificadores simbólicos em um único classificador final. Neste trabalho propomos o algoritmo GAESC-SG, um algoritmo genético que evolui classificadores simbólicos em um único classificador, utilizando operadores de crossover e mutação e também operadores de especialização e generalização. Os resultados obtidos nos experimentos realizados foram considerados interessantes, pois o GAESC-SG apresenta comportamento mais estável quanto ao número de regras e de gerações em relação ao GAESC .*

1. Introdução

Em problemas nos quais é necessário realizar predições de classificação, são normalmente utilizados algoritmos de aprendizado de máquina supervisionados. A um algoritmo de aprendizado supervisionado, é fornecido um conjunto de exemplos rotulados para o algoritmo induzir um classificador, cujo objetivo é rotular novos exemplos. Em muitos problemas do mundo real, é importante avaliar a precisão do classificador, mas também é im-

portante analisar o conhecimento nele contido, com o objetivo de descobrir novo conhecimento e/ou fornecer explicação da classificação dada. Um exemplo são os problemas de mineração de dados, nos quais extrair conhecimento é um dos principais objetivos do processo de mineração. Nesse caso, é indicado o uso de algoritmos de aprendizado simbólico, nos quais o modelo induzido pode ser diretamente interpretado pelo usuário/especialista do domínio. Em muitos domínios, os classificadores simbólicos apresentam taxas de acerto menores que classificadores induzidos por outros tipos de algoritmos de aprendizado. Uma solução é utilizar Algoritmos Genéticos — AGs — para evoluir classificadores em um único classificador final, com melhor poder de predição que os classificadores iniciais. Em [Bernardini et al. 2008], foi proposto um algoritmo genético para esse fim, denominado GAESC — *Genetic Algorithm for Evolving Symbolic Classifiers* —, que utiliza medidas de avaliação de regras de conhecimento [Lavrac et al. 1999] juntamente com medidas de avaliação de classificadores para avaliar a qualidade dos classificadores evoluídos. Os resultados obtidos nos experimentos realizados foram satisfatórios e promissores. Entretanto, experimentos com um número maior de conjuntos de dados eram necessários. Ainda, o estudo de outros dois operadores — operadores de generalização e especialização de classificadores — foi previsto como trabalho futuro para o GAESC. Assim, neste trabalho, propomos uma variação do GAESC, o GAESC-SG, que utiliza esses dois operadores além dos operadores clássicos de cross-over e mutação, e descrevemos experimentos realizados tanto com o GAESC quanto com o GAESC-SG utilizando 5 (cinco) conjuntos de dados naturais distintos.

Este trabalho está dividido como segue: Na Seção 2 são descritos conceitos de Aprendizado de Máquina Simbólico, importantes para uma boa compreensão deste trabalho. Na Seção 3 são descritos alguns trabalhos relacionados encontrados na literatura. Na Seção 4, é descrito o algoritmo genético GAESC. Na Seção 5, é descrito o algoritmo GAESC-SG, proposto neste trabalho. Na Seção 6, são descritos os experimentos realizados e os resultados obtidos. Por fim, na Seção 6 são feitas as considerações finais deste trabalho.

2. Aprendizado de Máquina Simbólico

Um conjunto de treinamento T é um conjunto de N exemplos rotulados $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, tendo sido utilizada para o rotulamento alguma função desconhecida $y = f(\mathbf{x})$. Os exemplos $\mathbf{x}_i$ são tipicamente vetores da forma $(x_{i1}, x_{i2}, ..., x_{im})$, cujos valores componentes são valores discretos ou contínuos, denominados atributos. Assim, x_{ij} denota o valor do j-ésimo atributo do exemplo $\mathbf{x}_i$. Em problemas de classificação, os valores y_i se referem a um conjunto discreto de N_{Cl} classes, *i.e.* $y_i \in \{C_1, C_2, ..., C_{N_{Cl}}\}$.

Dado um conjunto T de exemplos de treinamento, um algoritmo de aprendizado induz um *classificador* $\mathbf{h}$, o qual é uma hipótese sobre a verdadeira função f. Dado um novo exemplo $\mathbf{x}$, $\mathbf{h}$ prediz o valor correspondente y. Neste trabalho consideramos *classificadores simbólicos* aqueles cuja descrição pode ser representada por um conjunto de N_R regras não-ordenadas, *i.e.* $\mathbf{h} = \{R_1, R_2..., R_{N_R}\}$. O termo *não-ordenado* significa que cada regra pode ser interpretada isoladamente, ou seja, sem levar em consideração as outras regras do conjunto.

Uma *regra* R assume a forma **if** B **then** H ou, simbolicamente, $B \rightarrow H$, onde H é a cabeça, ou *conclusão*, da regra, e B é o corpo, ou *condição*, da regra. O corpo consiste

de uma disjunção de conjunções de testes de atributos na forma X_i *op Valor*, onde X_i é o nome do atributo, *op* é um operador do conjunto $\{=, \neq, <, \leq, >, \geq\}$ e *Value* é um valor válido do atributo X_i. Em uma *regra de classificação*, a cabeça H assume a forma *class* $= C_i$, onde $C_i \in \{C_1, ..., C_{N_{Cl}}\}$.

Dada uma regra $R = B \rightarrow H$ e um conjunto de exemplos T, denominamos $\mathcal{B} \subset T$ o conjunto de exemplos que satisfaz B e $\mathcal{H} \subset T$ o conjunto de exemplos que satisfaz H. Também, denominamos $\overline{\mathcal{B}}$ e $\overline{\mathcal{H}}$ os respectivos complementos dos conjuntos. Exemplos que satisfazem ambos B e H são *cobertos corretamente* pela regra R, e pertencem ao conjunto $\mathcal{B} \cap \mathcal{H}$. Exemplos que satisfazem B mas não H são *cobertos incorretamente* por R, e pertencem ao conjunto $\mathcal{B} \cap \overline{\mathcal{H}}$. Por outro lado, exemplos que não sastisfazem B *não são cobertos* por R, e pertencem ao conjunto $\overline{\mathcal{B}}$. Uma maneira de avaliar a qualidade de uma regra é calculando sua matriz de contingência [Lavrac et al. 1999], como pode ser visto na Tabela 1, onde a cardinalidade de um conjunto A é denominado a, *i.e.* $a = |A|$. Assim, b e h na Tabela 1 denotam o número de exemplos pertencentes aos conjuntos $\mathcal{B}$ e $\mathcal{H}$ respectivamente, *i.e.* $b = |\mathcal{B}|$ e $h = |\mathcal{H}|$, e assim sucessivamente. Da matriz de contingência de R, diversas medidas de qualidade de regras podem ser calculadas [Lavrac et al. 1999]. Neste trabalho nós usamos $Acc(R) = hb/b$, $Lap(R) = (bh+1)/(b+N_{Cl})$ e $Cov(R) = b$.

Tabela 1. Matriz de Contingência de uma Regra $B \rightarrow H$

	B	$\overline{B}$	
H	bh	$\overline{b}h$	h
$\overline{H}$	$b\overline{h}$	$\overline{bh}$	$\overline{h}$
	b	$\overline{b}$	n

Como mais de uma regra pode cobrir mais de uma instância, um conjunto de regras pode ser usado de diferentes maneiras para constituir um classificador. Um deles, denominado *método de classificação baseado em uma única regra (Single Rule — SR)*, consiste em escolher, de todas as regras que cobrem o exemplo a ser classificado, a melhor regra segundo algum critério de qualidade de regra. Como critério de qualidade, pode ser utilizada uma medida de qualidade de regra, como as mencionadas anteriormente. Em [Bernardini et al. 2008] foram descritos alguns experimentos realizados com o método de classificação SR, cujos resultados não foram considerados promissores e, assim, esse método não foi utilizado neste trabalho. Outro método, denominado *método de classificação baseado em múltiplas regras (Multiple Rule — MR)*, considera a combinação de todas as regras que cobrem o exemplo. Neste trabalho, MR foi implementado como segue: Considere MR_{C_i} o conjunto de regras que cobrem um exemplo $\mathbf{x}$ a ser classificado, e possuem a mesma cabeça, ou seja, *class* $= C_i$. Para efeito de simplificação, dizemos que o conjunto MR_{C_i} é da classe C_i. Para cada MR_{C_i}, é computada a medida de qualidade utilizada para o exemplo $\mathbf{x}$. Daí, é calculada a média $\widehat{\mu}_{MR_{C_i}}$ dos valores computados da medida de qualidade utilizada para cada conjunto MR_{C_i}. O exemplo é então classificado de acordo com a classe C_i cujo $\widehat{\mu}_{MR_{C_i}}$ oferece o melhor valor segundo a medida de qualidade utilizada.

Finalmente, a performance de um conjunto de regras como um classificador pode ser avaliada utilizando uma matriz de confusão, comparando a classificação dada pelo classificador em relação à classificação verdadeira dos exemplos do conjunto de teste. Para um problema binário, ou seja, com duas classes, geralmente chamadas positiva $+$ e

negativa $-$, a matriz de confusão é mostrada na Tabela 2, onde: $T_P(T_N)$ é o número de exemplos positivos (negativos) que são corretamente classificadas; e $F_P(F_N)$ é o número de exemplos negativos (positivos) classificados incorretamente. Assim como para avaliar as regras individualmente, há diversas medidas de qualidade de classificadores baseadas na matriz de confusão que podem ser calculadas. Neste trabalho, usamos as seguintes medidas: $Acc(\mathbf{h}) = \frac{T_P+T_N}{N}$; $F_1(\mathbf{h}) = \frac{2T_P}{2T_P+F_N+F_P}$; and $Prec(\mathbf{h}) = \frac{T_P}{T_P+F_P}$.

Tabela 2. Matriz de Confusão para Problemas de Classificação Binários

Classe	Predito +	Predito -
Verdadeiro +	T_P	F_N
Verdadeiro -	F_P	T_N

3. Algoritmos Genéticos para Evolução de Classificadores Simbólicos

Um Algoritmo Genético (AG) necessita de uma estrutura de dados que codifica uma solução para um problema que se deseja resolver — uma representação para um ponto no espaço de busca. Tal estrutura é denominada *genoma* e tal ponto no espaço é denominado *cromossomo*. A união de um cromossomo com sua aptidão é denominada *indivíduo*. Um parâmetro codificado no cromossomo é denominado *gene*. Dados um espaço de busca, uma função de aptidão, a qual determina quão boa a solução por ele representada é para a resolução de um problema, e um conjunto de operadores genéticos (seleção, *crossover* e mutação), o algoritmo genético clássico segue os seguintes passos [Michalewicz 1999]:

1. Gerar aleatoriamente uma população de n cromossomos.
2. Para cada cromossomo x da população, calcular a função de aptidão $f(x)$.
3. Repetir os seguintes passos até que n filhos tenham sido criados:
   (a) Selecionar um par de cromossomos da população corrente. A probabilidade de seleção do par de cromossomos é dada pela função de aptidão.
   (b) Aplicar a operação de *crossover* sobre os cromossomos selecionados.
   (c) Aplicar a operação de mutação em genes do filho criado e colocar o cromossomo resultante na nova população.
4. Substituir a população corrente pela nova população.
5. Voltar ao passo 2 se o(s) critério(s) de parada não for(em) satisfeito(s).

A principal motivação para o uso de algoritmos genéticos na descoberta de regras de conhecimento é a busca global no espaço do problema. Ainda, são capazes de lidar com a interação entre atributos, quando comparado aos algoritmos gulosos de aprendizado de regras [Freitas 2002]. Há vários trabalhos que variam as técnicas de representação e evolução, dentre os quais podemos citar [de la Iglesia et al. 2003, Riquelme et al. 2003, Zhu and Guan 2004, Halavati et al. 2009]. Esses trabalhos são baseados em duas estratégias básicas para evolução de regras e métodos híbridos que combinam as características positivas dessas duas estratégias. Tais estratégias são denominadas Michigan e Pittsburgh. A principal característica da estratégia Michigan [Holland 1986] é a evolução de regras individualmente, e a principal característica da estratégia Pittsburgh é evoluir conjuntos de regras. Os algoritmos genéticos citados anteriormente utilizam codificação binária para codificar as regras e, ainda, os classificadores iniciais geralmente são cromossomos (classificadores) formados por genes aleatórios.

Apesar de os algoritmos de aprendizado de regras gulosos serem menos eficientes que os algoritmos genéticos propostos, o objetivo desses algoritmos é realizar uma busca gulosa sobre o espaço de hipóteses da função verdadeira f e encontrar uma hipótese aproximada de f. Os algoritmos de aprendizado têm a característica de utilizar heurísticas distintas para realizar essa busca. As propostas do GAESC e do GAESC-SG utilizam regras construídas por esses algoritmos de indução de regras para inicialização dos indivíduos iniciais da população, o que pode tornar a busca do AG mais eficiente.

4. O Algoritmo GAESC

Em [Bernardini et al. 2008], foi proposto um algoritmo genético, denominado GAESC (*Genetic Algorithm for Evolving Symbolic Classifiers*), para evolução de classificadores simbólicos. O GAESC segue basicamente os passos do AG clássico. O que varia no GAESC em relação ao AG clássico, descrito na seção anterior, são seus componentes, descritos a seguir.

Codificação dos Indivíduos: No GAESC, cada gene de um indivíduo (classificador) é uma regra, e cada indivíduo é, portanto, uma seqüência (conjunto) de regras. Sendo assim, esse AG difere do AG binário clássico, que codifica os indivíduos com valores binários. Na população inicial do AG proposto, podem participar: (a) classificadores induzidos por diversos algoritmos de aprendizado simbólico, tais como $\mathcal{CN}2$, $\mathcal{C}4.5$ e $\mathcal{C}4.5rules$, ou outros algoritmos; (b) classificadores constituídos por conjunto de regras fornecidos pelo usuário; ou (c) classificadores construídos pelo sistema computacional que implementa o AG, descrito adiante. Para construir tais classificadores, o GAESC seleciona regras de uma base de regras, ilustrada na Figura 1. Essa base é formada por W regras que participam dos classificadores descritos nos itens anteriores. Na Figura 2 é ilustrado um indivíduo codificado, ou seja, um classificador contendo as regras 01, 10, 15, 09 e 07 da base de regras.

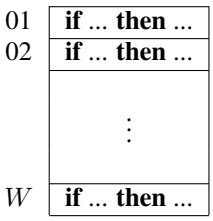


Figura 1. Base de Regras de Conhecimento

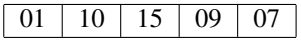


Figura 2. Exemplo de um indivíduo codificado

Operadores Genéticos: São dois os operadores genéticos utilizados no GAESC: *crossover* e mutação. O operador de *crossover* utilizado é o *crossover* assimétrico. Em um *crossover* assimétrico, a cada indivíduo da população corrente é associado aleatoriamente um valor entre 0 e 1. Os indivíduos, cujo valor associado é menor que a probabilidade de *crossover* p_c, fornecido pelo usuário, são selecionados para a operação de *crossover*. Para cada par de indivíduos "pais" são escolhidas duas posições, uma em cada indivíduo. Os indivíduos pais são divididos em dois segmentos conforme as posições escolhidas. Por fim, são criados os indivíduos filhos por meio da composição dos segmentos dos indivíduos pai. A operação de mutação é utilizada de maneira usual: para aplicar a operação de mutação, seleciona-se um gene de um indivíduo (aleatoriamente) e troca-se a regra selecionada por outra selecionada da base de regras disponível.

Funções de Avaliação: Como os indivíduos são classificadores, a função de avaliação deve avaliar seu comportamento em um conjunto de dados de teste. Ainda, como o método interno de classificação de um classificador pode variar, a função de avaliação do GAESC é dada em função dos dois métodos: o método de avaliação de classificadores HQ utilizando uma medida de avaliação de classificadores, combinada ao método de classificação MR utilizando uma medida de avaliação de regras. As medidas de avaliação implementadas que formam as funções HQ são $HQ_{Acc}(\mathbf{h}) = Acc(\mathbf{h})$, $HQ_{Prec}(\mathbf{h}) = Prec(\mathbf{h})$ e $HQ_{F1}(\mathbf{h}) = F1(\mathbf{h})$. Conforme mencionado na Seção 2, somente o método MR foi avaliado, e assim métodos MR propostos e utilizados neste trabalho são MR_{Acc}, MR_{Lap} e MR_{Cov}. Como a função de avaliação do GAESC é definida como sendo dependente do método de avaliação HQ e do método de classificação MR, as funções de avaliação possíveis de serem escolhidas são $MR_{Acc}HQ_{AC}$, $MR_{Cov}HQ_{Acc}$, $SR_{Lap}HQ_{F1}$, e assim sucessivamente.

Critério de Parada: Há 3 (três) critérios de parada possíveis de serem utilizados pelo GAESC [Bernardini et al. 2008], os quais são: (a) Execução de um número máximo de iterações dado pelo usuário; (b) Um método de convergência, ou seja, dado um número de gerações N_{gen}, se a função de avaliação não melhora nas últimas N_{gen} gerações, o GAESC pára; e (c) outro método de convergência similar ao método anterior, porém o GAESC pára se a média dos valores da função de avaliação de todos os indivíduos da população não melhora nas últimas N_{gen} gerações. Em [Bernardini et al. 2008], os melhores resultados obtidos foi utilizando o terceiro critério de parada, o qual foi utilizado neste trabalho

5. O Algoritmo GAESC-SG

O algoritmo GAESC-SG contém, além dos operadores e das características do algoritmo GAESC, outros dois operadores, um de generalização e um de especialização de classificadores:

Operador de Generalização de Classificador: Um indivíduo (classificador) pode ser generalizado adicionando-se uma regra a ele. Essa operação generaliza um conjunto de regras (classificador) porque o conjunto de regras é uma disjunção de regras. Assim, quando uma regra é inserida no classificador, um disjunto (regra) está sendo adicionado ao conjunto, o que em geral tende a aumentar o número de exemplos cobertos pelo classificador.

Operador de Especialização de Classificador: Ao contrário da generalização, um conjunto de regras (classificador) pode ser especializado removendo uma regra do conjunto de regras.

Em [Freitas 2002] esses dois operadores foram mencionados como possíveis operadores, porém não é de nosso conhecimento trabalhos que os tenham utilizado.

Para adicioná-los ao GAESC, ambos os operadores foram adicionados após a aplicação dos operadores de *crossover* e mutação. Para ambos os operadores, foi definida uma probabilidade de generalização p_g e especialização p_s. O Algoritmo 1 ilustra os passos do algoritmo GAESC-SG. Deve ser observado que o conjunto de treinamento é utilizado para (i) construir os classificadores iniciais; e (ii) calcular a função f nas linhas 3 e 12 do Algoritmo 1. O conjunto de validação é utilizado somente para podar o classificador final — linha 16 do Algoritmo 1.

Algoritmo 1 O Algoritmo GAESC-SG

Require: $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$: Conjunto de exemplos utilizado para construir os conjuntos de treinamento, teste e validação;
$C_{init} = \{\mathbf{h}_a, \mathbf{h}_b..., \mathbf{h}_q\}$;
$\mathcal{W} = \{R_1, ..., R_W\}$: Base de regras de conhecimento;
N_{ind}: Número de indivíduos na população;
Parâmetros de *cross-over*, mutação, especialização e generalização — p_c, p_m, p_g e p_s;
$f(\mathbf{h})$: função de avaliação dos indivíduos

1: Gerar a população inicial $P_{current} = \{\mathbf{h}_1, ..., \mathbf{h}_{N_{ind}}\}$ where $C_{init} \subset P_{current}$;
2: **for all** $\mathbf{h} \in P_{current}$ **do**
3: Calcular $f(\mathbf{h})$ usando o conjunto de treinamento;
4: **end for**
5: **while** Critério de parada não for satisfeito **do**
6: $\mathbf{h}$ = melhor_indivíduo($P_{current}$);
7: $P_{new} = \mathbf{h} + N_{ind} - 1$ indivíduos selecionados de $P_{current}$;
8: Aplicar operador de *cross-over* aos indivíduos selecionados de P_{new} utilizando p_c;
9: Aplicar operador de mutação aos indivíduos selecionados de P_{new} utilizando p_m;
10: Aplicar operador de generalização aos indivíduos selecionados de P_{new} utilizando p_g;
11: Aplicar operador de especialização aos indivíduos selecionados de P_{new} utilizando p_s;
12: Calcular f para os indivíduos modificados em P_{new} usando o conjunto de treinamento;
13: $P_{current} = P_{new}$;
14: **end while**
15: $\mathbf{h}$ = melhor_indivíduo($P_{current}$);
16: $\mathbf{h}^*$ =pós_processa($\mathbf{h}$); {Podar $\mathbf{h}$ usando o conjunto de validação}
17: **return** $\mathbf{h}^*$; {Classificador evoluído pelo AG}

6. Experimentos Realizados

Foram realizados experimentos utilizando 5 (cinco) bases de dados — Autos, Balance[1], Heart, Ionosphere e Tic-Tac-Toe — disponíveis em [Frank and Asuncion 2010]. Na Tabela 3 são descritas as principais características desses conjuntos de dados: número de exemplos — #Exs.; número de atributos contínuos e discretos #Atrib (continuous, discrete); distribuição dos exemplos nas classes — Class %; erro majoritário (Err. Maj.); presença ou ausência de valores desconhecidos — Val. Desc.?; e número de exemplos duplicados ou conflitantes — #Exs C/D. O comportamento do GAESC-SG foi avaliado variando as funções de avaliação e a utilização dos operadores de generalização e especialização.

Os experimentos foram realizados em 4 (quatro) cenários de experimentação: (i) GAESC: utilizando o GAESC; (ii) GAESC+G: utilizando o GAESC-SG somente com o operador de generalização — $p_g = 0.10$ e $p_g = 0.0$; (iii) GAESC+S: utilizando o GAESC-SG somente com o operador de especialização — $p_g = 0.0$ e $p_g = 0.10$; e (iv) GAESC+SG: utilizando o GAESC-SG com ambos os operadores — $p_g = 0.10$ e $p_g = 0.10$. Em todos os experimentos, os parâmetros de mutação e *cross-over* foram utilizados, respectivamente, $p_m = 0.01$ e $p_c = 0.40$. Os experimentos foram realizados utilizando 10-*fold cross-validation*. O conjunto de treinamento foi utilizado para construir regras de associação para classificação utilizando o algoritmo Apriori (os atributos contínuos foram discretizados somente para a construção das regras) e três classificado-

[1]O conjunto de dados original denominado Balance-Scale foi modificado removendo os exemplos da classe B, pois esta classe possuía somente 7,84% do número total de exemplos, o que introduz um problema de desbalanceamento de classes [Batista et al. 2004].

Dataset	#Exs	#Atrib (cont.,disc.)	Class (%)	Err. Maj.	Val. Desc?	#Exs C/D	
Autos	205	25 (15,10)	safe (44.88%) risky (52.12%)	44.88% in risky	Y	0 (0.00%)	
Balance	576	4 (0,4)	L (46.08%) R (53.92%)	53.92% in L	N	0 (0.00%)	
Heart	270	13 (5,8) (5,8)	1 (55.56%) 2 (44.44%)	44.44% in 1	Y	0 (0.00%)	
Ionosphere	351	33 (33,0)	good bad	64.10% 35.90%	35.90% em good	N	1 (0.28%)
Tic-tac-toe	958	9 (0,9)	positive negative	65.34% 34.66%	34.66% em positive	N	0 (0.00%)

Tabela 3. Datasets description

res simbólicos foram construídos utilizando 2 (dois) algoritmos clássicos de aprendizado simbólico — $\mathcal{CN}2$ e $\mathcal{C}4.5$. Todas as regras induzidas foram utilizadas para compor a base de regras de conhecimento. A população inicial consistiu de dois classificadores induzidos pelos algoritmos $\mathcal{CN}2$ (regras não-ordenadas) e $\mathcal{C}4.5$ (C_{init}), e os outros $N_{ind} - C_{init}$ foram construídos utilizando regras selecionadas aleatoriamente da base de regras. O número de regras dos $N_{ind} - C_{init}$ indivíduos da população inicial foi definido como o número médio de regras dos dois classificadores de C_{init}.

Nós usamos a média da taxa de erro, calculada utilizando 10-*fold cross-validation*, como métrica para avaliar nossa abordagem. Para analisar se há diferença entre os métodos de classificação de exemplos MR, entre os métodos de avaliação de classificadores HQ e entre os algoritmos GAESC e GAESC-SG, foi executado o teste estatístico Friedman[2]. O teste Friedman foi executado com quatro diferentes hipóteses nulas: (H_1) na qual os resultados utilizando os algoritmos $\mathcal{C}4.5$, $\mathcal{CN}2$ e GAESC são comparáveis; (H_2) na qual os resultados utilizando os três métodos de classificação MR_{Acc}, MR_{Lap} ou MR_{Cov} são comparáveis; (H_3) na qual os resultados utilizando os três métodos de avaliação de classificadores $HQ_{Acc}(\mathbf{h})$, $HQ_{Prec}(\mathbf{h})$ ou $HQ_{F1}(\mathbf{h})$ são comparáveis; e (H_4) na qual os resultados utilizando GAESC ou GAESC-SG são comparáveis. Quando a hipótese nula é rejeitada pelo teste Friedman, podemos proceder um teste *post-hoc* para detectar quais diferenças entre os métodos são significantes. Nós executamos o teste Nemenyi para detectar as diferenças significativas.

O teste Friedman rejeitou as hipóteses nulas H_1, H_2 e H_3 com 95% de confiança. Nas Figuras 3, 4(a) e 4(b) são mostrados os resultados dos testes Nemenyi executados. Para cada gráfico, a linha mais fina sobre o gráfico indica o intervalo de diferença crítica (veja [Demšar 2006]), diferença essa que indica quando dois métodos estatisticamente apresentam comportamentos distintos.

Na Figura 3, podemos observar que somente o algoritmo GAESC apresenta diferença significativa em relação ao $\mathcal{C}4.5$, porém o mesmo não ocorre com o $\mathcal{CN}2$. Entretanto, quando observamos as taxas de erro do $\mathcal{CN}2$, $\mathcal{C}4.5$ e GAESC, observamos

[2]O teste Friedman é o teste não-paramétrico equivalente à ANOVA. Veja [Demšar 2006] para uma discussão sobre testes estatísticos para aprendizado de máquina.

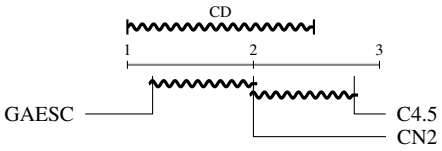
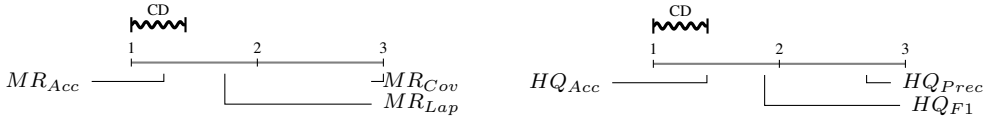


Figura 3. Testes de Hipótese H_1: Os algoritmos $\mathcal{CN}2$, $\mathcal{C}4.5$ e GAESC apresenta comportamentos semelhantes?



(a) Critérios MR de classificação (b) Critérios HQ avaliação

Figura 4. Testes de Hipótese H_2 e H_3

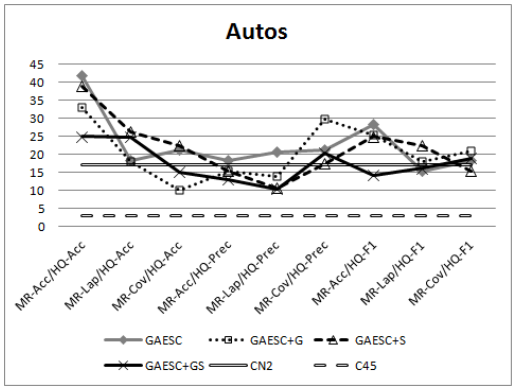
diferenças notáveis para alguns conjuntos de dados, indicando que ainda assim o GAESC é bastante promissor se comparado aos dois algoritmos de aprendizado. Na Figura 4(a), podemos observar que o método de classificação que apresentam melhores resultados segundo os testes executados é o MR_{Acc}, e na Figura 4(b), podemos observar que o método de avaliação que oferece melhores resultados é o HQ_{Acc}. Em relação à hipótese nula H_4, o teste Friedman não rejeitou a hipótese nula, indicando que o comportamento dos algoritmos GAESC e GAESC-SG apresentam comportamentos semelhantes.

Analisamos, também, o comportamento dos algoritmos GAESC e GAESC-SG, nos quatro cenários de experimentação, quanto ao número médio de regras do indivíduo evoluído e o número médio de gerações do algoritmo genético. Nas Figuras 5(a) a 5(e) é mostrado o número médio de regras, calculado a partir dos classificadores construídos nas 10 iterações do 10-*fold cross-validation*, dos classificadores evoluídos pelo GAESC e pelo GAESC-SG nos quatro cenários, variando a função de avaliação, para cada conjunto de dados. Já nas Figuras 6(a) a 6(e) é mostrado o número médio de gerações, calculado a partir dos classificadores construídos nas 10 iterações do 10-*fold cross-validation*, dos classificadores evoluídos pelo GAESC e pelo GAESC-SG nos quatro cenários, variando a função de avaliação, para cada conjunto de dados.

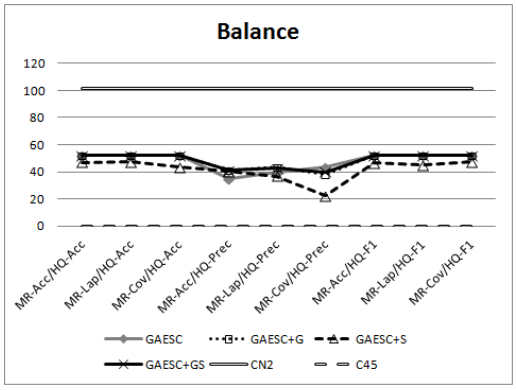
Mesmo não havendo diferença significativa entre o GAESC e o GAESC-SG segundo o teste Friedman considerando a medida de erro dos classificadores evoluídos, observamos que, em relação ao número médio de gerações e ao número médio de regras, nos cenários de experimentação GAESC+SG há uma tendência em manter uma média entre as diferentes execuções do GAESC, o que pode indicar que o GAESC-SG apresenta um comportamento mais estável em relação ao GAESC.

7. Considerações Finais

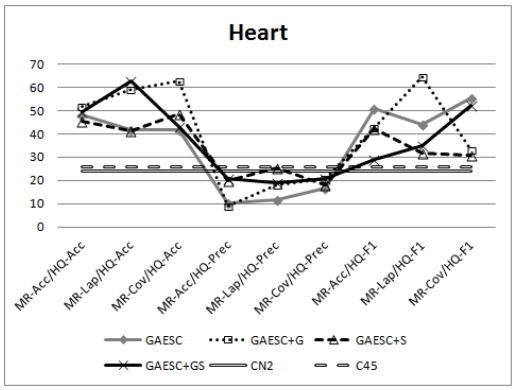
Neste trabalho, foi descrito o algoritmo GAESC, proposto em [Bernardini et al. 2008], bem como foi proposto o algoritmo GAESC-SG, uma extensão do GAESC. Uma vantagem desses algoritmos genéticos em relação aos algoritmos genéticos tradicionais para evolução de conjuntos de regras é a inicialização dos indivíduos, que se dá com regras induzidas de algoritmos de aprendizado, o que torna mais rápido o processo de evolução



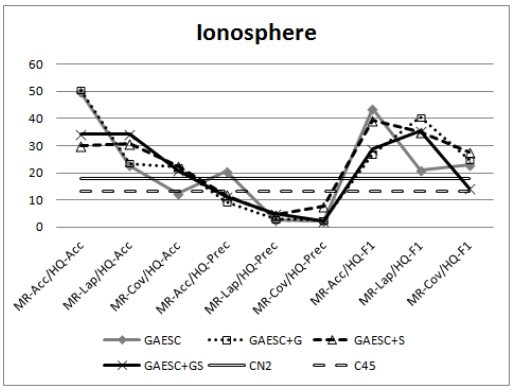
(a) Conjunto de Dados Autos



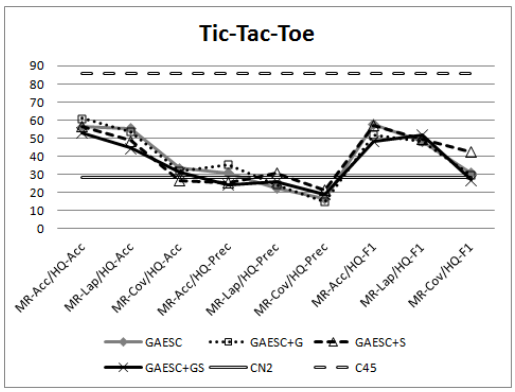
(b) Conjunto de Dados Balance



(c) Conjunto de Dados Heart

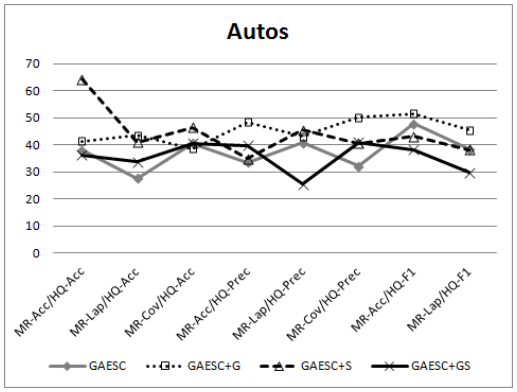


(d) Conjunto de Dados Ionosphere

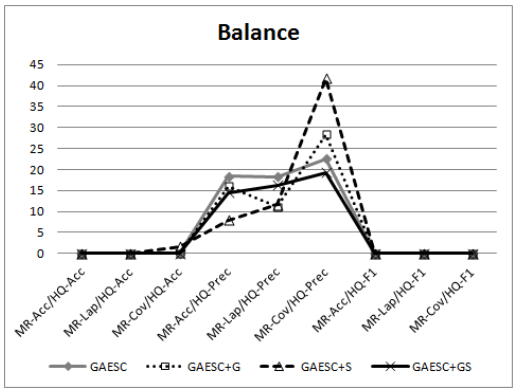


(e) Conjunto de Dados Tic-Tac-Toe

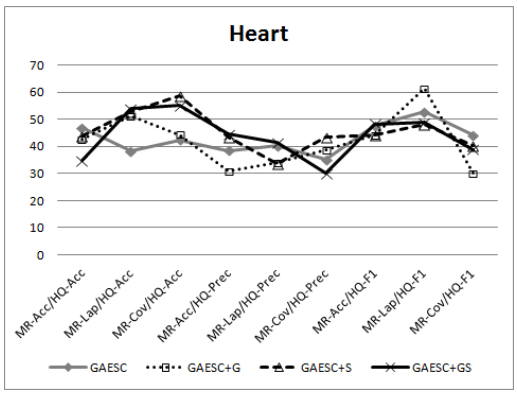
Figura 5. Número Médio de Regras dos Classificadores Evoluídos para os 5 (Cinco) Conjuntos de Dados em todos os Cenários de Experimentação



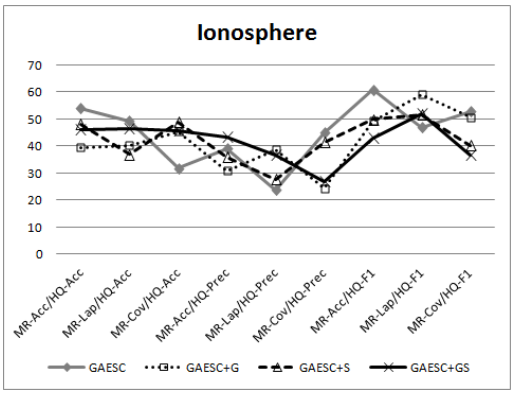
(a) Conjunto de Dados Autos



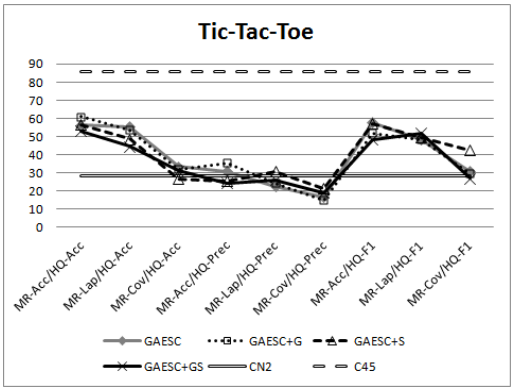
(b) Conjunto de Dados Balance



(c) Conjunto de Dados Heart



(d) Conjunto de Dados Ionosphere



(e) Conjunto de Dados Tic-Tac-Toe

Figura 6. Número Médio de Gerações Realizadas por Cada Algoritmo Genético para os 5 (Cinco) Conjuntos de Dados em todos os Cenários de Experimentação

dos classificadores. Os resultados obtidos foram considerados interessantes, pois o ganho no poder de predição do GAESC e do GAESC-SG em relação aos classificadores induzidos pelos algoritmos de aprendizado utilizado é estatisticamente significante e, ainda, em relação ao algoritmo GAESC-SG, há um ganho de performance quanto ao número de regras do classificador final evoluído e também no número de gerações, se comparado ao algoritmo GAESC. Como trabalhos futuros, pretendemos explorar a evolução de classificadores simbólicos para aprendizado *on-line*.

Agradecimentos

Os autores agradecem ao Prof. Ronaldo Cristiano Prati (UFABC) por suas valiosas observações e contribuições, a Jean Metzz (USP), pelo auxilio para execução dos testes de hipótese, e aos revisores por suas contribuições para a melhoria deste trabalho.

Referências

Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29.

Bernardini, F. C., Monard, M. C., and Prati, R. C. (2008). Evolving sets of symbolic classifiers into a single symbolic classifier using genetic algorithms. In *International Conference on Hybrid Intelligent Systems — HIS 2008*, volume 1, Barcelona, Espanha.

de la Iglesia, B., Philpott, M., Bagnall, A., and Rayward-Smith, V. (2003). Data mining rules using multi-objective evolutionary algorithms. In *Proceedings of IEEE Congress on Evolutionary Computations*, volume 3, pages 1552–1559.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository.

Freitas, A. A. (2002). *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer Verlag.

Halavati, R., Shouraki, S., Lotfi, S., and Esfandiar, P. (2009). Symbiotic evolution of rule based classifier systems. *International Journal on Artificial Intelligence Tools (IJAIT)*, 18(1):1–16.

Holland, J. H. (1986). *Machine Learning: An Artificial Intelligence Approach*, volume 2, chapter Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems.

Lavrac, N., Flach, P., and Zupan, B. (1999). Rule evaluation measures: a unifying view. In *Proc. 9th International Workshop on Inductive Logic Programming. Lecture Notes in Artificial Intelligence*, volume 1634, pages 74–185. Springer Verlag.

Michalewicz, Z. (1999). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag.

Riquelme, J., Toro, J., and Aguilar-Ruiz, M. (2003). Evolutionary learning of hierarchical decision rules. *IEEE Transactions on Systems, Man, and Cybernetics*, 33(2):324–334.

Zhu, F. and Guan, S. (2004). Ordered incremental training with genetic algorithms. *Int. Journal of Intelligent Systems*, 19(12):1239–1256.