Otimização da AUC via Redes Neurais Evolutivas para Classificação de Dados Desbalanceados

Johnathan M. Melo Neto, Cristiano L. Castro, Antônio P. Braga

Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Engenharia Elétrica
Av. Pres. Antônio Carlos, 6627 - Pampulha
Belo Horizonte, MG, Brasil
Emails: jmmn.mg@gmail.com, crislcastro@ufmg.br, apbraga@ufmg.br

Resumo A classificação de dados cujas classes se apresentam em proporções desbalanceadas é um problema relativamente comum de se encontrar na prática, porém de difícil resolução com o uso dos tradicionais métodos baseados na minimização da taxa de erro. Diversas métricas têm sido propostas para contornar este desafio, como a área abaixo da curva ROC (AUC), cujo objetivo é realizar o correto ranqueamento das instâncias sem a necessidade de se especificar um limiar de discriminação. Sendo a AUC uma função não diferenciável, os métodos de aprendizagem baseados em direção de busca, tal como *backpropagation*, fazem uso de uma aproximação diferenciável desta métrica, o que pode levar a resultados subótimos. Este artigo propõe uma rede neural evolutiva que otimiza diretamente a AUC. Desta forma, sua expressão exata pode ser utilizada, dado que algoritmos evolutivos não exigem a diferenciabilidade da função objetivo, em contraste com os usuais algoritmos baseados no cálculo da derivada. Tal metodologia visa fornecer classificadores com melhor desempenho, pois confia na formulação exata da AUC. A estratégia se mostrou promissora nas simulações realizadas, apresentando desempenho superior às abordagens comparadas, obtendo melhor equilíbrio de classificação em relação às taxas de acerto das classes.

Palavras-chave: redes neurais evolutivas, otimização, área abaixo da curva roc, classificação, desbalanceamento

1 Introdução

O atual cenário da pesquisa em aprendizado de máquina apresenta diversos desafios relacionados ao problema de classificação com dados desbalanceados. Uma vasta gama de aplicações lida com dados naturalmente gerados com distribuição não uniforme entre classes, como por exemplo, diagnóstico clínico, detecção de fraude, e classificação textual, em que a predominância quantitativa de uma classe sobre a outra é nítida. Em grande parte dos problemas desta natureza, as instâncias que constituem a classe minoritária são aquelas de interesse na classificação. Entretanto, tal desbalanceamento força os classificadores tradicionais a

serem tendenciosos para a classe majoritária, o que pode levar a um desempenho insatisfatório de classificação das instâncias raras [8], sobretudo devido aos métodos de otimização empregados, geralmente baseados na minimização do erro global. Na literatura, três abordagens que visam solucionar este problema são comumente exploradas: 1) reamostragem dos dados; 2) aprendizado sensível ao custo; e 3) métodos híbridos que combinam as duas técnicas anteriores.

Uma abordagem alternativa que visa contornar o desbalanceamento das distribuições é baseada na maximização da área abaixo da curva ROC (AUC), cujos resultados têm apontado para uma significativa melhora de desempenho em cenários desbalanceados [9]. Todavia, a AUC é uma função não-diferenciável, e as metodologias que utilizam redes neurais artificiais (RNAs) para otimizá-la frequentemente confiam em uma função objetivo cuja expressão é uma aproximação diferenciável, portanto inexata, da AUC. Tal abordagem permite utilizar os clássicos métodos de otimização baseados em direção de busca (gradiente), porém, essa estratégia de aproximação pode levar a resultados subótimos [2].

Este artigo propõe uma RNA que otimiza diretamente a AUC através do uso de um algoritmo evolutivo (AE). Desta forma, a expressão exata da AUC pode ser utilizada, dado que AEs não exigem a diferenciabilidade da função objetivo, em contraste com os usuais algoritmos baseados no cálculo da derivada. Tal metodologia visa fornecer classificadores com melhor desempenho, uma vez que confia na formulação exata da AUC. Ademais, a abordagem evolutiva aqui utilizada foca tanto na busca local quanto global por soluções ótimas ao longo do espaço de busca, através da adaptação dos parâmetros de mutação ao longo das gerações. A evolução das redes se dá na topologia e nos pesos de suas conexões.

O artigo está organizado como descrito a seguir. A Seção 2 apresenta o conceito de Redes Neurais Evolutivas e apresenta a metodologia proposta para otimizar diretamente a AUC. A Seção 3 contém a descrição dos experimentos realizados. A Seção 4 apresenta os resultados e a discussão. Finalmente, a Seção 5 conclui o trabalho deste artigo.

2 Rede Neural Artificial Evolucionária Adaptativa

A computação evolucionista aplicada na otimização de RNAs vem sendo explorada por sua capacidade de busca global ao longo do espaço de busca de parâmetros da rede (pesos, camadas, número de neurônios, etc.). Comumente denominada de redes neurais evolutivas (RNEs), tal abordagem reduz o volume de ajustes manuais para o projeto da rede, gerando RNAs de diferentes arquiteturas e pesos durante o processo evolutivo, consequentemente evitando a típica estagnação em ótimos locais dos métodos determinísticos. Diversos trabalhos [3,6,7] se dedicaram ao desenvolvimento de classificadores baseados em RNEs. O algoritmo proposto por [7], fonte de inspiração deste artigo, implementa uma RNE que evolui simultaneamente a topologia e os pesos da rede, e utiliza técnica de mutação adaptativa durante o processo evolutivo visando garantir o equilíbrio entre busca global e local.

Cabe mencionar, no entanto, que os trabalhos de RNEs supracitados não foram projetados para lidar com o problema de desbalanceamento de classes, pois possuem funções de aptidão baseadas na taxa de erro global de classificação. O presente artigo introduz uma abordagem alternativa capaz de mitigar o efeito do desbalanceamento no desempenho de classificação através do uso da AUC na avaliação dos indivíduos. O trabalho desenvolvido por [5] possui objetivos correlatos, contudo utiliza arquitetura pré-definida para as redes, o que reduz o espaço de busca por melhores soluções. Com base no algoritmo HEANN [7], os detalhes do método aqui proposto estão descritos a seguir.

2.1 Esquema de Codificação

A rede implementada é do tipo *feedforward* arbitrariamente conectada (ACN), ilustrada na Fig. 1. Ela contém N_i entradas, que dependem do problema de classificação a ser tratado; N_h neurônios escondidos, parâmetro definido pelo projetista; e apenas uma saída, dado que o método está focado na otimização de classificadores binários. Cada ACN representa uma solução candidata a ser evoluída no algoritmo de treinamento. A função de ativação sigmoidal é utilizada em todos os nós. O esquema de codificação da rede possui duas estruturas de dados: o vetor de nós e a matriz de conexão (Fig. 2). Os valores da matriz correspondem aos pesos das conexões entre os neurônios das colunas e os neurônios das linhas. O peso $p_{1,3}$, por exemplo, é o peso da conexão que associa os neurônios I_1 e H_3. Os elementos da diagonal principal são os valores de *bias* dos neurônios. O vetor de nós é preenchido com booleanos associados aos neurônios da camada escondida, representando sua presença (1) ou ausência (0) na rede. O neurônio H_5, por exemplo, é inexistente na rede, e portanto todos os seus pesos associados estão vazios na matriz de conexão. Esta codificação permite a evolução tanto da topologia da rede quanto dos pesos.

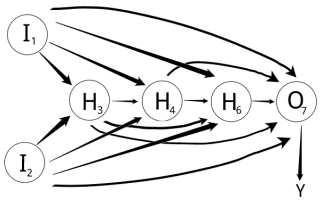


Figura 1: Rede Neural *Feedforward* Arbitrariamente Conectada. Os símbolos referenciam palavras da língua inglesa (*input*, *hidden*, *output*), indicando os nós de cada camada (entrada, escondida, saída).

2.2 Função de Aptidão

A função de aptidão F utilizada nesta RNE é dada pela soma ponderada da AUC (F_{auc}) e da complexidade da rede (F_{net}), como mostra a Eq. 1, em que

$$\begin{array}{cccc} H_3 & H_4 & H_5 & H_6 \\ \left(\begin{array}{cccc} 1 & 1 & 0 & 1 \end{array} \right) \end{array} \qquad \begin{array}{c} \\ I_1 \\ I_2 \\ H_3 \\ H_4 \\ H_5 \\ H_6 \\ O_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} I_1 & I_2 & H_3 & H_4 & H_5 & H_6 & O_7 \end{array} \\ \left(\begin{array}{ccccccc} & & & & & & \\ & & & & & & \\ p_{1,3} & p_{2,3} & bias & & & & \\ p_{1,4} & p_{2,4} & p_{3,4} & bias & & & \\ & & & & & & \\ p_{1,6} & p_{2,6} & p_{3,6} & p_{4,6} & & bias & \\ p_{1,7} & p_{2,7} & p_{3,7} & p_{4,7} & & p_{6,7} & bias \end{array} \right) \end{array}$$

Figura 2: Esquema de codificação da RNA. Vetor de Nós (esq.) e Matriz de Conexão (dir.).

as constantes α e β são pré-definidas, e a avaliação de F é feita com base no conjunto de treinamento. Como se prioriza a maximização da AUC, α tende a possuir um peso mais relevante que β, frequentemente encontrados na literatura na proporção 1:9 ou 2:8. Indivíduos mais aptos possuem maiores valores de F, portanto trata-se de um problema de maximização.

$$F = \alpha F_{auc} + \beta F_{net} \tag{1}$$

A AUC é uma medida adequada para o cenário de classificação binária desbalanceada [1], e sua definição discreta é dada por

$$F_{auc}(f) = \frac{1}{PQ}(\sum_{j=1}^{P}\sum_{k=1}^{Q} g(f(\overrightarrow{x_j^+}) - f(\overrightarrow{x_k^-}))) \ , \tag{2}$$

sendo P e Q o total de instâncias positivas e negativas, respectivamente; $\overrightarrow{x^+}$ e $\overrightarrow{x^-}$ os vetores de instâncias positivas e negativas, respectivamente; $f(x_i)$ a saída do classificador f para o exemplo i; e $g(z)$ definida como

$$g(z) = \begin{cases} 0 & \text{se } z < 0 \\ 0.5 & \text{se } z = 0 \\ 1 & \text{se } z > 0 \ . \end{cases} \tag{3}$$

Finalmente, F_{net} é definida como $1/\ln C$, sendo C o número de conexões da rede. Minimizar a complexidade da rede parte do pressuposto que redes muito complexas, além de gastarem computacionalmente mais tempo para serem processadas, tendem ao sobre-ajuste dos dados caso o algoritmo de aprendizagem não possua mecanismos que garantam boa capacidade de generalização.

2.3 Índice de Perda de Generalização

Para garantir que a evolução dos indivíduos ocorra sem sobre-ajuste dos dados (*overfitting*), é utilizado o índice de perda de generalização, $GL(t)$, definido como

$$GL(t) = \left(1 - \frac{AUC_{best}(t) + 1}{AUC_{best}(t-1) + 1}\right) \ , \tag{4}$$

sendo $AUC_{best}(t)$ o maior valor de AUC da geração atual t, e $AUC_{best}(t-1)$ o maior valor de AUC da geração $t-1$. Ambos os valores são obtidos sobre um conjunto de validação, dado que a avaliação de instâncias não utilizadas pelo algoritmo de treinamento permite observar o comportamento do classificador em cenários desconhecidos até então, dando maior confiabilidade ao índice em questão. Valores positivos de GL indicam que não houve melhora nas soluções ao longo da evolução, sugerindo possível sobre-ajuste da população ou dificuldade de busca local. Por outro lado, valores negativos de GL indicam progressiva melhora das soluções atuais em relação às anteriores, o que pressupõe que a busca global está caminhando para regiões mais promissoras do espaço de busca [7]. Este indicador é utilizado tanto como critério de parada quanto como parâmetro da mutação adaptativa, descrita a seguir.

2.4 Estratégia de Mutação Adaptativa

A técnica de mutação adaptativa objetiva alcançar o equilíbrio entre busca glocal e local no espaço de busca por soluções ótimas. Tal equilíbrio é controlado através da: 1) probabilidade de mutação da estrutura da rede (adição/remoção de neurônios da camada escondida); 2) probabilidade de mutação dos pesos; e 3) magnitude da pertubação aplicada nos pesos que sofreram mutação. Pequenos valores destes parâmetros favorecem a busca local, e altos valores a busca global. Neste sentido, duas formas de se calcular as probabilidades de mutação e a magnitude da pertubação são feitas.

A primeira forma, que atinge igualmente todos os indivíduos, é baseada no índice GL. As probabilidades de mutação global da estrutura (P_{ge}) e dos pesos (P_{gp}), e a magnitude da pertubação global (M_g) na geração t são dadas por

$$P_{ge}(t) = tmpP_{ge} \cdot \frac{s(t)}{S} \ , \qquad P_{gp}(t) = tmpP_{gp} \cdot \frac{s(t)}{S} \ ,$$

$$M_g(t) = tmpM_g \cdot \frac{s(t)}{S} \tag{5}$$

sendo S uma constante que determina o número máximo de gerações consecutivas que podem surgir sem que as soluções criadas apresentem alguma melhora. Quando não há melhoria das soluções atuais em relação às anteriores ($GL \geq 0$), o contador $s(t)$, que inicialmente tem valor igual a S, decrementa uma unidade, sugerindo possível sobre-ajuste da população ou dificuldade de busca local, diminuindo gradativamente a participação das componentes globais. Em contrapartida, quando houver aprimoramento das soluções atuais em relação às anteriores ($GL < 0$), pressupõe-se que a busca global está caminhando para regiões mais promissoras do espaço de busca, portanto $s(t)$ retorna para seu valor inicial S, e as variáveis temporárias $tmpP_{ge}$, $tmpP_{gp}$, e $tmpM_g$ são atualizadas com os

valores atuais de P_{ge}, P_{gp} e M_g, respectivamente. Tais variáveis temporárias determinam a taxa de decaimento das componentes globais, refinando a busca global sempre que houver melhora nas soluções, pois tais valores são reduzidos gradativamente ao longo da evolução.

A segunda forma, que atinge cada indivíduo de forma particular, é baseada nos valores de aptidão de cada solução. As probabilidades de mutação local da estrutura (P_{le}) e dos pesos (P_{lp}), e a magnitude da pertubação local (M_l) do n-ésimo indivíduo são dadas por

$$P_{le}(n) = baseP_{le} \cdot \left(1 - \frac{F(n)}{F_{best}}\right) \ , \qquad P_{lp}(n) = baseP_{lp} \cdot \left(1 - \frac{F(n)}{F_{best}}\right) \ ,$$
$$M_l(n) = baseM_l \cdot \left(1 - \frac{F(n)}{F_{best}}\right) \tag{6}$$

sendo F_{best} o melhor valor de aptidão da população atual, e $F(n)$ o valor de aptidão do n-ésimo indivíduo. As constantes $baseP_{le}$, $baseP_{lp}$, e $baseM_l$ são definidas pelo projetista e controlam os máximos valores permitidos para as taxas de mutação e para a magnitude da pertubação. Neste sentido, a busca local será maior em indivíduos menos aptos, e quanto maior a aptidão de um indivíduo, menor será sua busca local, objetivando a preservação dos mesmos.

Dadas as equações acima, as probabilidades finais de mutação estrutural (P_e) e dos pesos (P_p), e a magnitude final da pertubação dos pesos (M) do indivíduo n da geração t são definidas como a soma de suas componentes global e local:

$$P_e(n,t) = P_{ge}(t) + P_{le}(n) \ , \qquad P_p(n,t) = P_{gp}(t) + P_{lp}(n) \ ,$$
$$M(n,t) = M_g(t) + M_l(n) \ . \tag{7}$$

2.5 Critério de Parada

O critério de parada é atingido quando: 1) $s(t)$ chega a zero, indicando a não obtenção de melhores indivíduos durante S gerações consecutivas; ou 2) quando o número máximo de gerações preestabelecido é alcançado.

2.6 Mecanismo de Seleção

Realizada a mutação em todos os indivíduos da população, o filho menos adaptado é substituído pelo indivíduo pai mais apto da geração anterior, caso este já não esteja presente na população. Esta solução pai e o restante das soluções filhas são selecionadas para a próxima geração. Percebe-se, portanto, o caráter elitista e diversificado do mecanismo de seleção: elitista pois sempre mantém consigo o melhor indivíduo de todas as gerações, e diversificado porque seleciona indistintamente todos os filhos (exceto o menos apto) para a próxima geração, evitando que super indivíduos dominem a população e se estacionem prematuramente em uma bacia de máximo local.

2.7 Ciclo Geracional

Cada rede da população inicial é gerada preenchendo o vetor de nós e a matriz de conexão com valores aleatórios uniformes. O vetor de nós é preenchido com 0s e 1s, e a matriz de conexão com números reais entre -1 e 1. Os parâmetros de mutação global, por utilizarem o índice GL, necessitam de um intervalo de tempo para verificar se houve perda de generalização no processo evolutivo, e por isso são atualizados a cada 10 gerações. Com relação aos operadores de variação, se a probabilidade P_e for satisfeita em alguma posição do vetor de nós ou da matriz de conexão, ocorrerá mutação: no vetor, tal posição sofre *bit flipping*; e na matriz, tal peso é criado ou removido. Caso a probabilidade de mutação dos pesos P_p for satisfeita em algum peso, este sofre adição de pertubação gaussiana com média zero e desvio padrão M. Calcula-se GL com base nos dados de validação. Caso haja melhora da geração atual em relação às anteriores ($GL < 0$), a rede que possuir melhor AUC em relação aos dados de validação é armazenada como melhor RNA encontrada. Caso haja empate entre a melhor RNA da geração atual e a melhor RNA das gerações passadas ($GL = 0$), a rede com maior AUC em relação aos dados de treinamento é armazenada. Persistindo o empate, a RNA com menor número de conexões é a escolhida. Finalmente, caso haja piora das soluções ($GL > 0$), a melhor rede neural não é atualizada. Em seguida, o critério de parada é analisado, podendo encerrar a execução do algoritmo, ou selecionar os próximos pais e continuar o ciclo geracional. Ao finalizar o algoritmo, a rede armazenada como melhor é utilizada para classificar os dados de teste.

3 Metodologia Experimental

A RNE adaptativa aqui implementada (doravante denominada HEANN-AUC) foi comparada com: 1) RNA *backpropagation* baseada na minimização da taxa de erro (BP-ERRO); 2) RNA *backpropagation* baseada na maximização da aproximação diferenciável da AUC (BP-AUC); e 3) RNE com função objetivo baseada na minimização da taxa de erro (HEANN-ERRO). O algoritmo BP-ERRO foi implementado utilizando a *Neural Network Toolbox* do MATLAB, no qual os pesos e *bias* da rede são atualizados de acordo com o gradiente descendente. O classificador BP-AUC foi retirado de [1], e o método HEANN-ERRO utiliza a RNE aqui desenvolvida, com função objetivo baseada na taxa de erro.

Com exceção dos pesos da função de aptidão (Eq. 1), os demais valores dos parâmetros utilizados pela HEANN-AUC proposta foram sugeridos por [7]. O tamanho da população é de 100 indivíduos, que podem evoluir ao longo de, no máximo, 3000 gerações. Cada rede pode ter um número máximo de 10 neurônios na camada escondida. Os pesos α e β são 0.95 e 0.05, respectivamente. Os valores relacionados às probabilidades de mutação adaptativa da estrutura da rede ($baseP_{le}$, $tmpP_{ge}$) e dos pesos ($baseP_{lp}$, $tmpP_{gp}$) é 0.02, e os relacionados à magnitude da pertubação dos pesos ($baseM_l$, $tmpM_g$) é 2. Por fim, o parâmetro S possui valor 500.

As métricas utilizadas para avaliação e comparação dos classificadores são: AUC, taxa de positivos verdadeiros (TPR), taxa de negativos verdadeiros (TNR),

acurácia, e *G-mean*. A métrica *G-mean* corresponde à média geométrica entre TPR e TNR, e mede o desempenho equilibrado de um classificador em relação às taxas de acerto de ambas as classes [1]. Os algoritmos baseados na AUC, originalmente, não estabelecem um limiar de decisão, pois o objetivo dos mesmos não é discriminar as instâncias, mas ranqueá-las. Para avaliar a TPR, TNR, *G-mean*, e acurácia destes métodos, um limiar de decisão deve ser estabelecido. Neste trabalho, o valor mais próximo do ponto $(0, 1)$ da curva ROC é definido como limiar, dado que este ponto tende a apresentar boa capacidade de classificação para ambas as classes.

Foram selecionadas 10 bases de dados do repositório UCI [4] com diferentes graus de balanceamento entre classes, sendo os atributos de cada base normalizados linearmente para o intervalo $[0, 1]$. A Tab. 1 descreve cada base, apresentando a proporção de balanceamento, isto é, a razão entre o número de instâncias da classe minoritária e o número total de instâncias. Algumas bases possuíam originalmente mais de duas classes e, portanto, foram reduzidas para problemas de classificação binária, rotulando uma única classe como positiva, e todas as restantes como negativa. Foram escolhidas, propositalmente, bases de diversos graus de balanceamento a fim de se avaliar o desempenho dos classificadores diante de problemas de naturezas distintas. Tais bases foram particionadas em três subdivisões: treinamento, validação, e teste. O conjunto de teste representa 30% de toda a base e não participa do processo de otimização, sendo utilizado apenas para avaliar a capacidade de generalização do método. Os outros 70% dos dados são divididos em treinamento e validação, ambos participando diretamente do processo de evolução das redes.

A reamostragem dos dados de treinamento e validação foi executada mediante o esquema de validação cruzada *k-fold*. O valor empiricamente escolhido foi $k = 3$ por apresentar melhor desempenho geral das métricas. Em cada conjunto de dados, a proporção de classes positivas e negativas foi mantida, a fim de evitar o surgimento de subconjuntos (*folds*) sem a presença da classe minoritária. A melhor rede gerada pela validação cruzada é escolhida para classificar o conjunto de teste. Para cada método de classificação, 50 execuções independentes foram simuladas para cada uma das bases, e a cada execução, todas as instâncias foram reordenadas aleatoriamente, evitando assim a divisão dos mesmos dados da rodada anterior, reforçando o caráter generalista do classificador. Por fim, a média e o desvio padrão das métricas geradas pelas 50 execuções foram calculadas.

4 Resultados e Discussão

A Tab. 2 apresenta os resultados experimentais obtidos para os dados de teste. Os valores em negrito destacam os maiores valores obtidos para cada métrica. O índice *G-mean*, por ser redundante em relação aos valores de TPR e TNR, não foi incluído na tabela. Uma representação visual dos resultados é ilustrada pela Fig. 3 com os gráficos comparativos das médias de cada métrica em função do grau de balanceamento dos dados. Pela Fig. 3a, pode-se notar que, para as bases mais desbalanceadas, há grande discrepância entre os valores de AUC

Tabela 1: Informações sobre as bases de dados selecionadas para os experimentos.

Base de dados	# atributos	# positivas	# negativas	Grau de balanceamento [%]
Abalone (19 vs. all) [**a19**]	8	32	4145	0.8
Yeast (5 vs. all) [**y5**]	8	51	1433	3.4
Yeast (9 vs. 1) [**y9-1**]	8	20	463	4.1
Abalone (18 vs. 9) [**a18-9**]	8	42	689	5.7
Vowel (0 vs. all) [**vow**]	10	90	900	9.1
Glass (7 vs. all) [**gls**]	9	29	185	13.6
SPECTF Heart [**hrt**]	44	55	212	20.6
WP Breast Cancer [**wpbc**]	33	47	151	23.7
German Credit [**gmn**]	24	300	700	30.0
Pima Indians Diabetes [**pid**]	8	268	500	34.9

apresentados pelos métodos baseados na AUC (∼85%) e os métodos de otimização baseados no erro (∼50%). Os métodos baseados na AUC obtiveram melhor desempenho de ranqueamento em praticamente todas as bases. À medida que o grau de balanceamento aumenta, contudo, os valores dos quatro métodos tendem a convergir (∼80%). As métricas TPR e TNR ajudam a entender a razão da má qualidade de ranqueamento dos métodos baseados no erro para as bases fortemente desbalanceadas. No caso da TPR (Fig. 3c), percebe-se que os métodos BP-ERRO e HEANN-ERRO obtiveram valores muito próximos de zero, e para a TNR (Fig. 3d) o índice alcançou praticamente 100%. Tal fato indica que, no intuito de minimizar o erro de classificação, os algoritmos rotularam praticamente todas as instâncias como negativas. Dado que o número de instâncias positivas é muito pequeno para bases fortemente desbalanceadas, a acurácia de classificação gerada foi alta (Fig. 3b), satisfazendo o objetivo dos otimizadores, contudo não adquirindo inteligência suficiente para discriminar corretamente as classes positivas. É neste sentido que os algoritmos baseados na AUC se destacam, uma vez que obtiveram relativamente altos valores de TPR (∼80%) e TNR (∼84%) mesmo nas bases altamente desbalanceadas (de grau 3.4%, por exemplo), refletindo, desta forma, em uma acurácia de classificação satisfatória (∼83%), o que sugere que, ao lidar com problemas desbalanceados, os métodos baseados na AUC são mais recomendados.

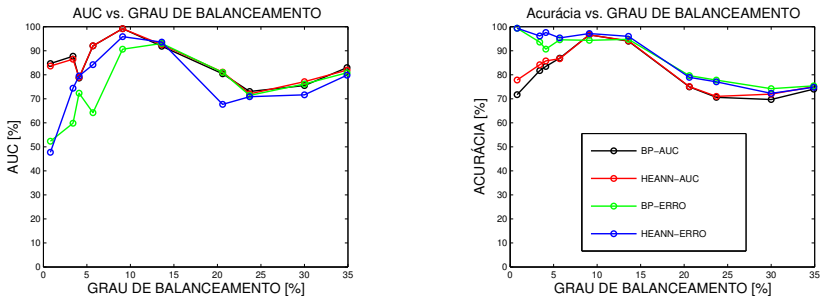
Já para as bases mais balanceadas, nota-se que tanto a AUC como a acurácia convergiram em todos os métodos. Entretanto, os valores de TPR e TNR apresentaram diferenças significativas. Para o grau de balanceamento de 34.9%, por exemplo, os algoritmos baseados no erro obtiveram maior TNR (∼87%) e menor TPR (∼53%), uma vez que priorizam a rotulação das instâncias como classes negativas, que por serem predominantes, geram maior acurácia. Os algoritmos baseados na AUC, por outro lado, obtiveram valores próximos de TPR (∼78%) e TNR (∼73%), uma vez que não priorizam a rotulação das instâncias como classes positivas ou negativas, já que o objetivo de tais métodos é realizar um ranqueamento correto, não tendo compromisso com a maximização da acurácia.

Tabela 2: Comparação dos resultados apresentados pelos quatro classificadores para as dez bases de dados. Valores indicam média e desvio padrão das 50 execuções.

Base de dados	Algoritmo	AUC [%]	Acurácia [%]	TPR [%]	TNR [%]
a19	HEANN-AUC	83.63 ± 04.54	77.81 ± 05.70	$\mathbf{85 \pm 06}$	078 ± 06
	BP-AUC	$\mathbf{84.68 \pm 03.59}$	71.71 ± 03.74	81 ± 13	072 ± 04
	HEANN-ERRO	47.74 ± 11.89	$\mathbf{99.36 \pm 00.01}$	00 ± 00	$\mathbf{100 \pm 00}$
	BP-ERRO	52.33 ± 10.38	$\mathbf{99.36 \pm 00.01}$	00 ± 00	$\mathbf{100 \pm 00}$
y5	HEANN-AUC	86.48 ± 03.42	84.19 ± 03.80	$\mathbf{83 \pm 05}$	084 ± 04
	BP-AUC	$\mathbf{87.73 \pm 03.46}$	81.74 ± 03.68	77 ± 10	084 ± 04
	HEANN-ERRO	74.40 ± 13.31	$\mathbf{96.15 \pm 00.55}$	06 ± 06	099 ± 01
	BP-ERRO	59.85 ± 13.45	93.60 ± 00.55	00 ± 01	$\mathbf{100 \pm 00}$
y9-1	HEANN-AUC	78.93 ± 10.17	85.76 ± 11.42	$\mathbf{73 \pm 11}$	086 ± 12
	BP-AUC	78.62 ± 08.97	83.47 ± 06.78	59 ± 17	092 ± 08
	HEANN-ERRO	$\mathbf{79.51 \pm 09.87}$	$\mathbf{97.58 \pm 01.24}$	52 ± 19	099 ± 01
	BP-ERRO	72.32 ± 14.45	90.68 ± 01.99	38 ± 19	$\mathbf{100 \pm 00}$
a18-9	HEANN-AUC	$\mathbf{92.08 \pm 02.84}$	86.69 ± 04.12	$\mathbf{87 \pm 05}$	087 ± 04
	BP-AUC	92.02 ± 02.81	86.90 ± 02.66	80 ± 10	087 ± 03
	HEANN-ERRO	84.21 ± 06.73	$\mathbf{95.34 \pm 00.71}$	30 ± 10	099 ± 01
	BP-ERRO	64.23 ± 10.08	94.55 ± 00.19	01 ± 02	$\mathbf{100 \pm 00}$
vow	HEANN-AUC	99.18 ± 00.56	96.60 ± 01.76	$\mathbf{98 \pm 02}$	97 ± 02
	BP-AUC	$\mathbf{99.22 \pm 00.57}$	96.64 ± 01.92	95 ± 05	97 ± 02
	HEANN-ERRO	95.87 ± 02.45	$\mathbf{97.17 \pm 00.65}$	77 ± 07	$\mathbf{99 \pm 01}$
	BP-ERRO	90.64 ± 05.35	94.33 ± 00.93	47 ± 11	$\mathbf{99 \pm 01}$
gls	HEANN-AUC	92.54 ± 04.67	94.15 ± 02.82	$\mathbf{91 \pm 07}$	95 ± 03
	BP-AUC	91.87 ± 04.48	93.97 ± 02.28	87 ± 08	96 ± 02
	HEANN-ERRO	$\mathbf{93.58 \pm 05.19}$	$\mathbf{95.99 \pm 01.85}$	86 ± 10	97 ± 02
	BP-ERRO	93.04 ± 08.41	94.77 ± 02.25	78 ± 15	$\mathbf{98 \pm 02}$
hrt	HEANN-AUC	$\mathbf{81.11 \pm 03.81}$	75.09 ± 04.10	$\mathbf{82 \pm 05}$	73 ± 05
	BP-AUC	80.41 ± 04.56	74.96 ± 04.12	62 ± 13	79 ± 06
	HEANN-ERRO	67.68 ± 09.28	78.92 ± 01.86	11 ± 10	$\mathbf{95 \pm 03}$
	BP-ERRO	80.95 ± 03.62	$\mathbf{79.63 \pm 02.34}$	26 ± 09	93 ± 03
wpbc	HEANN-AUC	72.04 ± 05.66	70.99 ± 04.71	$\mathbf{71 \pm 07}$	71 ± 06
	BP-AUC	$\mathbf{73.03 \pm 05.61}$	70.60 ± 05.92	52 ± 11	78 ± 08
	HEANN-ERRO	70.87 ± 06.44	77.07 ± 02.60	27 ± 07	92 ± 03
	BP-ERRO	71.41 ± 07.78	$\mathbf{77.77 \pm 02.76}$	26 ± 10	$\mathbf{93 \pm 04}$
gmn	HEANN-AUC	$\mathbf{77.15 \pm 02.28}$	71.94 ± 02.55	$\mathbf{73 \pm 03}$	72 ± 04
	BP-AUC	75.52 ± 02.72	69.71 ± 02.25	66 ± 06	71 ± 03
	HEANN-ERRO	71.64 ± 02.86	72.28 ± 01.37	32 ± 08	$\mathbf{90 \pm 03}$
	BP-ERRO	76.16 ± 01.94	$\mathbf{74.20 \pm 01.55}$	44 ± 05	87 ± 02
pid	HEANN-AUC	81.93 ± 02.41	75.06 ± 02.46	77 ± 03	74 ± 04
	BP-AUC	$\mathbf{82.98 \pm 02.30}$	74.02 ± 02.96	$\mathbf{78 \pm 05}$	72 ± 05
	HEANN-ERRO	79.83 ± 02.74	74.80 ± 02.31	52 ± 06	$\mathbf{87 \pm 04}$
	BP-ERRO	80.82 ± 02.70	$\mathbf{75.37 \pm 02.22}$	54 ± 05	$\mathbf{87 \pm 02}$

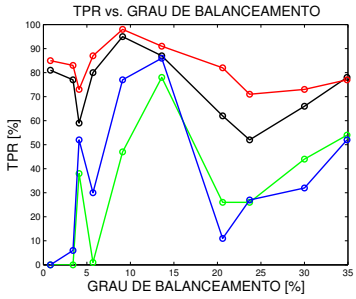
Tal afirmação é reforçada ao se observar os valores de *G-mean* dos quatro classificadores (Fig. 3e). Os algoritmos baseados na AUC tendem a gerar valores mais equilibrados de TPR e TNR, indicando taxas de acerto elevadas e equiparadas.

Para a avaliação do método proposto neste trabalho (HEANN-AUC), cabe aqui uma comparação com a RNA baseada na aproximação da AUC (BP-AUC), uma vez que ambos algoritmos têm por objetivo a maximização da capacidade de ranqueamento das instâncias. A Fig. 3f ilustra o gráfico comparativo das médias geométricas entre AUC e acurácia de ambos os métodos em função do grau de balanceamento dos dados, a fim de se avaliar tanto a capacidade de

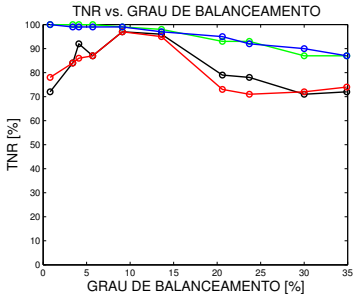


(a) AUC

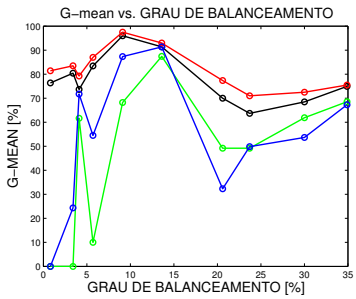
(b) Acurácia



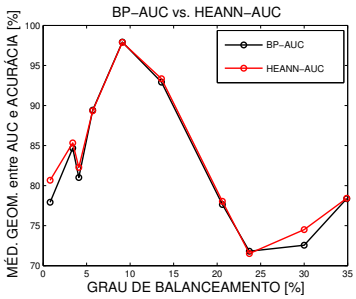
(c) TPR



(d) TNR



(e) G-mean



(f) $\sqrt{AUC \times Acuracia}$

Figura 3: Gráficos comparativos entre os métodos. Média das métricas de desempenho em função do grau de balanceamento dos dados.

ranqueamento quanto a capacidade de se classificar os dados sob um limiar de decisão. A HEANN-AUC obteve desempenho igual ou superior ao método BP-AUC para todas as bases de dados. Em relação à métrica *G-mean* (Fig. 3e), o método HEANN-AUC obteve melhor desempenho em todas as bases, indicando que este tende a possuir melhor equilíbrio de classificação em relação às taxas de acerto de ambas as classes.

Com relação ao custo computacional, os métodos HEANN-AUC e HEANN-ERRO demandaram tempo significativamente maior de processamento (∼5 a 10

h) para finalizar suas execuções se comparados com os algoritmos BP-AUC e BP-ERRO, que demandaram poucos minutos, o que caracteriza certa desvantagem dos métodos meta-heurísticos em relação aos determinísticos.

5 Conclusões

Neste trabalho foi apresentada uma metodologia para classificação e ranqueamento de instâncias utilizando rede neural artificial evolutiva adaptativa cujo objetivo principal é a maximização da área abaixo da curva ROC, através da evolução da topologia e dos pesos das redes. Os testes realizados com bases de dados de diferentes graus de balanceamento sugerem que a estratégia de se utilizar a formulação exata da AUC, em detrimento da aproximação diferenciável do método determinístico comparado, é bem sucedida e tende a gerar resultados ligeiramente mais significativos para as métricas utilizadas, principalmente para os índices TPR e *G-mean*. Ademais, nota-se nítida superioridade dos métodos baseados na maximização da AUC em relação aos algoritmos baseados na minimização do erro, sobretudo para as bases fortemente desbalanceadas. A abordagem adaptativa dos parâmetros de mutação mostrou-se eficiente nas buscas locais a globais ao longo do espaço de busca, e o critério de parada baseado na perda de generalização foi capaz de evitar sobre-ajuste dos dados, melhorando a capacidade de generalização para os conjuntos de teste.

Referências

1. Castro, C.L.: Novos critérios para seleção de modelos neurais em problemas de classificação com dados desbalanceados. Ph.D. thesis, Universidade Federal de Minas Gerais, Belo Horizonte (Out 2011)
2. Castro, C.L., Braga, A.P.: Optimization of the area under the roc curve. In: 2008 10th Brazilian Symposium on Neural Networks. pp. 141–146 (Oct 2008)
3. Kopel, A.: Neural Networks Performance and Structure Optimization Using Genetic Algorithms. Master's thesis, Faculty of California Polytechnic State University, San Luis Obispo (Ago 2012)
4. Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`
5. Lu, X., Tang, K., Yao, X.: Evolving Neural Networks with Maximum AUC for Imbalanced Data Classification, pp. 335–342. Springer Berlin Heidelberg, Berlin, Heidelberg (2010), `https://doi.org/10.1007/978-3-642-13769-3_41`
6. Mineu, N.L., da Silva, A.J., Ludermir, T.B.: Evolving neural networks using differential evolution with neighborhood-based mutation and simple subpopulation scheme. In: 2012 Brazilian Symposium on Neural Networks. pp. 190–195 (Out 2012)
7. Oong, T.H., Isa, N.A.M.: Adaptive evolutionary artificial neural networks for pattern classification. IEEE Transactions on Neural Networks 22(11), 1823–1836 (Nov 2011)
8. Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., Kennedy, P.J.: Training deep neural networks on imbalanced data sets. In: 2016 International Joint Conference on Neural Networks (IJCNN). pp. 4368–4374 (July 2016)
9. Zhi, Y., en Xia, G., dong Jin, W.: Optimizing area under the roc curve using genetic algorithm. In: Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on. vol. 1, pp. 672–675 (Jun 2011)