Aprendizado Semissupervisionado com Extreme Learning Machines e Matrizes de Afinidade

Leonardo Jose Silvestre*†, Antônio Pádua Braga*
*Programa de Pós-Graduação em Engenharia Elétrica -
Universidade Federal de Minas Gerais -
Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil
†Departamento de Computação e Eletrônica
Universidade Federal do Espírito Santo
Rodovia BR 101 Norte, Km. 60, 29932-540, São Mateus, ES, Brasil
Email: lsilvestre@ceunes.ufes.br

***Abstract*—This paper presents an approach that uses a supervised classification algorithm and information obtained from the data structure to perform semi-supervised classification. The structural information is used to insert a perturbation in the training of the supervised algorithm or for estimating labels to unlabeled data. The framework developed is instantiated with Extreme Learning Machines (ELMs) and affinity matrices. The results for real datasets (UCI repository), compared to the original ELM, indicate the valitidy of the method.**

Keywords*—*Extreme Learning Machines, Affinity Matrices, Semi-supervised Learning

***Resumo*—Este artigo apresenta uma abordagem que utiliza um algoritmo de classificação supervisionada e informações obtidas *a priori* da estrutura dos dados para realizar classificação semissupervisionada. A informação obtida *a priori* é utilizada para inserir perturbação no treinamento do algoritmo supervisionado ou para estimar rótulos para os padrões não-rotulados. Para instanciar o *framework* para aprendizado semissupervisionado desenvolvido são utilizadas Máquinas de Aprendizado Extremo (*Extreme Learning Machine* - ELMs) e matrizes de afinidade. Os resultados obtidos para bases de dados reais (repositório UCI), comparados aos da ELM original, apontam a validade do método.**

Palavras-chave*—*Máquinas de Aprendizado Extremo, Matrizes de Afinidade, Aprendizado Semissupervisionado

I. Introdução

O *aprendizado supervisionado* necessita de que os dados possuam rótulos, cuja obtenção pode ter um custo muito alto quando comparado a obter os próprios dados - por exemplo, gravar a fala é um processo que tem custo relativamente baixo, mas rotular corretamente a fala gravada pode ser custoso [1]. O *aprendizado não-supervisionado*, por sua vez, lida com a total ausência de rótulos e tem como desafios a determinação automática do número de agrupamentos existentes e determinação de a qual classe pertence cada agrupamento encontrado.

Como uma abordagem intermediária entre o aprendizado supervisionado e o aprendizado não-supervisionado, encontra-se o *aprendizado semissupervisionado*. Ele supõe a existência de um pequeno número de amostras rotuladas em meio a um grande número de amostras não-rotuladas, ou seja, o conjunto de padrões $\mathbf{D}$ é tal que $\mathbf{D}_L \cup \mathbf{D}_U$, onde $\mathbf{D}_L = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N_L}$ - dados rotulados - e $\mathbf{D}_U = \{\mathbf{x}_j\}_{j=1}^{N_U}$ - dados não-rotulados, e $N_U \gg N_L$. Assim, além de utilizar informações do pequeno conjunto rotulado $\mathbf{D}_L$, um algoritmo de aprendizado semissupervisionado visa a considerar também informações importantes fornecidas pelo conjunto não-rotulado $\mathbf{D}_U$.

Para induzir o modelo com um conjunto de dados rotulados escasso, os algoritmos de aprendizado semissupervisionados necessitam de alguma informação adicional. Essa informação é, usualmente, obtida *a priori* da estrutura dos dados. Um tipo de informação estrutural que pode ser útil para a construção de modelos semissupervisionados é aquele fornecido por matrizes de afinidade, que expressam as relações entre as amostras e entre grupos de amostras presentes nos conjuntos de dados rotulados e não rotulados.

No contexto das RNAs, as Máquinas de Aprendizado Extremo (*Extreme Learning Machines* -ELMs) [2], [3] têm atraído a atenção da comunidade acadêmica, principalmente por sua simplicidade, rapidez de treinamento e boa capacidade de generalização. O treinamento de uma ELM ocorre em duas etapas: primeiramente, é feita uma projeção aleatória dos padrões na camada escondida e, posteriormente, os pesos da camada de saída são calculados de forma analítica, utilizando a inversa generalizada de Moore-Penrose [4].

O fato de as ELMs terem seu treinamento realizado em duas etapas permite visualizar uma forma de adaptá-las para classificação semissupervisionada, tendo em vista que é possível interferir no treinamento, inserindo informações estruturais dos dados. A abordagem adotada neste trabalho explora esse conceito, apresentando um *framework* para classificação semissupervisionada que utiliza algoritmos de classificação supervisionada e informações estruturais obtidas *a priori* dos dados, e instanciando esse *framework* com ELMs e Matrizes de Afinidade.

O restante deste artigo é organizado como segue: na Seção II é apresentada a revisão bibliográfica sobre ELMs, Aprendizado Semissupervisionado e Matrizes de Afinidade. Na Seção III, é apresentado o método desenvolvido, com as suas três diferentes versões. A Seção IV apresenta os resultados obtidos e, por fim, a Seção V apresenta as discussões e conclusões.

II. Revisão Bibliográfica

A. Extreme Learning Machines

As Máquinas de Aprendizado Extremo - *Extreme Learning Machines* (ELMs) [2], [3] - utilizam o conceito de projeção aleatória para treinar RNAs de uma única camada. Ao contrário do treinamento tradicional de RNAs, o treinamento das ELMs não envolve ajuste iterativo dos parâmetros. Os pesos e termos de polarização da camada escondida recebem valores aleatórios segundo uma distribuição uniforme, e os pesos de saída são calculados por meio da inversa generalizada - pseudo-inversa. É importante destacar que a camada escondida de uma ELM possui mais neurônios do que uma rede treinada, por exemplo, com o algoritmo de Retro Propagação de erros (*Error Back Propagation* - EBP), pois a projeção em um espaço de mais alta dimensão tem o objetivo de linearizar o problema: pelo Teorema de Cover [5], "um problema complexo de classificação de padrões disposto não linearmente em um espaço de alta dimensão tem maior probabilidade de ser linearmente separável do que em um espaço de baixa dimensionalidade." [6].

Assim, considere - sem perda de generalidade, pois a ELM possui capacidade para realizar classificação multiclasses -, para um problema de classificação binária, um conjunto de N padrões de treinamento $\mathbf{X}_{N\times n}$, com n sendo o número de atributos, e a saída desejada desses padrões, $\mathbf{Y}_{N\times 1}$. A matriz de pesos da camada escondida (com p neurônios), $\mathbf{Z}_{n\times p}$, é inicializada com valores aleatórios. A saída $\mathbf{H}_{N\times p}$ da camada escondida é dada, então, por:

$$\mathbf{H} = \Psi(\mathbf{X}, \mathbf{Z}), \tag{1}$$

onde $\Psi(\cdot\,,\cdot)$ é uma função de ativação que pode ou não ser diferenciável.

A matriz de pesos da camada de saída, $\mathbf{W}_{p\times 1}$, é calculada por meio da pseudo-inversa de $\mathbf{H}$, $\mathbf{H}^{\dagger}$:

$$\mathbf{W} = \mathbf{H}^{\dagger}\mathbf{Y}. \tag{2}$$

A partir o cálculo da matriz $\mathbf{W}$, para encontrar a saída da rede, faz-se:

$$\hat{\mathbf{Y}} = \mathbf{H}\mathbf{W}. \tag{3}$$

O erro de treinamento pode ser calculado por:

$$\mathbf{E} = \mathbf{Y} - \hat{\mathbf{Y}}. \tag{4}$$

A saída $\hat{\mathbf{Y}}_{test}$ para o conjunto de padrões de teste $\mathbf{X}_{test}$ pode, então, ser obtida:

$$\hat{\mathbf{Y}}_{test} = \mathbf{H}_{test}\mathbf{W}, \tag{5}$$

onde $\mathbf{H}_{test}$ é a projeção dos padrões de teste na camada escondida:

$$\mathbf{H}_{test} = \Psi(\mathbf{X}_{test}\mathbf{Z}). \tag{6}$$

B. Aprendizado Semissupervisionado

O aprendizado semissupervisionado considera o conjunto $\mathbf{D}$ de padrões como sendo dividido entre $\mathbf{D}_L = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N_L}$ - dados rotulados - e $\mathbf{D}_U = \{\mathbf{x}_j\}_{j=1}^{N_U}$ - dados não-rotulados. Muitas vezes, tem-se que $N_U \gg N_L$. $\mathbf{D}_U$ e $\mathbf{D}_L$ devem ter sido gerados a partir da mesma função de probabilidade.

Dado que o conjunto de dados rotulados é escasso, é necessário que o aprendizado semissupervisionado seja baseado em hipóteses, conhecidas como premissas do aprendizado semissupervisionado. As principais premissas são as que seguem [7] - os algoritmos baseiam-se em uma ou mais dessas premissas:

- ***Smoothness Assumption***: "Se dois pontos, x_1 e x_2 em uma região de alta densidade estão próximos, então também estarão próximas suas saídas correspondentes, y_1 e y_2", ou seja, se dois pontos estão próximos, é provável que eles tenham o mesmo rótulo.
- ***Clustering Assumption***: As instâncias em cada classe formam um grupo coerente, ou seja, se os pontos estão em um mesmo agrupamento, é provável que estejam em uma mesma classe, ou ainda, a fronteira de decisão encontra-se em uma região de baixa densidade. Este é um caso especial da assunção anterior.
- ***Manifold Assumption***: Os dados de mais alta dimensão estão dentro de um espaço de dimensão menor, ou seja, é possível reduzir a dimensão sem perda de informação.

1) Máquina de Aprendizado Extremo Semissupervisionada - Semi-Supervised ELM: Recentemente, foram desenvolvidas versões semissupervisionadas da ELM. A versão apresentada em [8], chamada de ELM Semissupervisionada - *Semi-supervised ELM* (SELM), é aplicada em localização baseada em redes wi-fi, e tem sua abordagem baseada na construção de um grafo semi-rotulado, a partir das amostras rotuladas e não-rotuladas. Cada amostra é representada em um vértice, o qual é conectado aos seus vizinhos. Baseado em [9], é definida, para a função f definida no grafo, uma função de suavização. Posteriormente, são calculados os pesos da camada de saída da ELM.

Já a versão apresentada em [10], também chamada de SELM, é baseada em *co-training*, o qual utiliza duas ELMs: uma é treinada com os dados rotulados e rotula algumas amostras não rotuladas para fornecer como amostras de treinamento para a outra. O processo é repetido por essa outra, até que um número k de iterações seja atingido. Um dos problemas do *co-training* é a geração de ruídos (amostras rotuladas incorretamente), mas a ELM, por lidar bem com ruídos, é uma boa candidata para *co-training*.

C. Matrizes de Afinidade

Uma matriz de afinidade (ou de similaridade ou, ainda, de proximidade) $\mathbf{A}_{N\times N}$, onde N é o número de padrões, é tal que cada elemento a_{ij} possui uma medida de afinidade entre os padrões $\mathbf{x}_i$ e $\mathbf{x}_j$. Para o caso de a afinidade ser reflexiva, tem-se que $a_{ij} = a_{ji}$, isto é, $\mathbf{A}$ é simétrica [11]. Um exemplo de medida de afinidade é a distância euclidiana entre os padrões. O produto escalar ou produto interno também é uma

medida de similaridade, em especial o produto interno *Kernel*, muito utilizado em Aprendizado de Máquina na construção de Máquinas de Vetor de Suporte (*Support Vector Machines* - SVMs) [12].

1) Matriz de Proximidade FCM: É possível obter matrizes de proximidade a partir de métodos de agrupamento. Em especial, o C-Médias Nebuloso - *Fuzzy C-Means* (FCM) [13] gera, como resultado, uma matriz $\mathbf{U}$ que contém o grau de pertinência de cada padrão a determinado grupo. A partir da matriz $\mathbf{U}$, pode-se obter cada elemento da matriz de proximidade $\mathbf{A}$ como segue [14]:

$$a_{kl} = \sum_{i=1}^{c} min(u_{ik}, u_{il}) \tag{7}$$

2) Kernels: Um *kernel* ou *produto interno kernel* pode ser definido por [6]:

$$K(\mathbf{x}, \mathbf{x}_i) = \varphi^T(\mathbf{x})\varphi(\mathbf{x}_i). \tag{8}$$

Um *kernel* é, portanto, "uma função K que mapeia os pontos no espaço de entrada de dimensão n_0 para pontos correspondentes em um novo espaço de dimensão n_1", conhecido como espaço de características. "A operação de mapeamento ou transformação consiste no produto interno dos pontos segundo uma função não linear implícita" [15].

Uma das principais funções utilizadas para a implementação de *kernel* é a gaussiana:

$$k(\mathbf{x}, \mathbf{x}_i) = e^{\frac{-\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}}, \tag{9}$$

onde σ^2 é a largura (raio) comum a todos os *kernels*, especificada *a priori* pelo usuário [6]. Um *kernel* gerado a partir da Equação 9 é conhecido como *kernel* RBF (Função de Base Radial - *Radial Basis Function*).

Para o caso de um produto interno simples, tem-se um *kernel* linear:

$$k(\mathbf{x}, \mathbf{x}_i) = \mathbf{x}\mathbf{x}_i \tag{10}$$

III. Método Proposto

Conforme introduzido na Seção I, o método desenvolvido visa a interferir no treinamento de um algoritmo de classificação supervisionada T para realizar classificação semissupervisionada, utilizando informações estruturais dos dados, obtidas *a priori* - tais como aquelas fornecidas por uma matriz de afinidade $\mathbf{A}$. Esse ajuste no treinamento é feito de duas formas principais: a primeira, para o caso de algoritmos de classificação supervisionada que realizam o treinamento em duas etapas (projeção dos padrões em um espaço de mais alta dimensão, por meio de uma transformação não-linear, seguida da estimação de uma separação linear nesse espaço), como a ELM, ocorre por meio da ponderação, com $\mathbf{A}$, da matriz resultante da projeção no espaço de mais alta dimensão, conforme apresentado pela Figura 1; já a segunda, mais abrangente por não impor condições ao tipo de treinamento de T, é feita por meio da estimação de rótulos para os padrões não rotulados, também utilizando $\mathbf{A}$. Uma terceira forma, que combina as duas anteriores, também é apresentada.

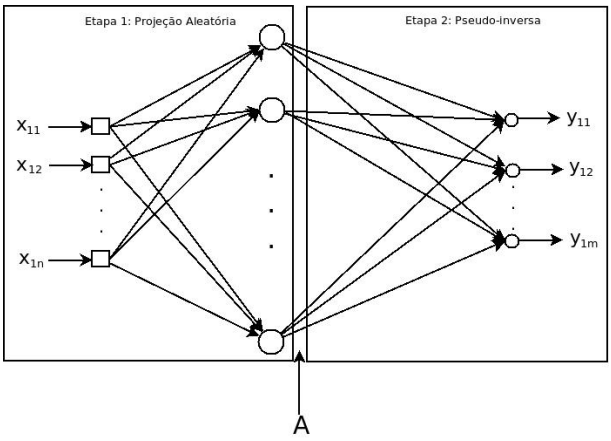


Figura 1: Inserção de perturbação - informação sobre estrutura dos dados - em uma ELM.

Seja $\mathbf{A}$ uma matriz de afinidade, conforme Seção II-C. Como visto, $\mathbf{A}$ é simétrica e cada elemento a_{kl} de $\mathbf{A}$ contém uma medida de afinidade entre os padrões $\mathbf{x}_k$ e $\mathbf{x}_l$. A matriz $\mathbf{S}$, que contém uma ponderação variável sobre $\mathbf{A}$, pode ser definida como:

$$\mathbf{S} = \lambda\mathbf{I} + (1 - \lambda)\mathbf{A}, \tag{11}$$

onde $\lambda \in [0, 1]$ e $\mathbf{I}$ é a matriz identidade, de dimensão $N \times N$, a mesma dimensão de $\mathbf{A}$ e, consequentemente, de $\mathbf{S}$. A variável λ é um termo que permite aumentar ou diminuir a importância de $\mathbf{A}$, ou seja, quando $\lambda = 0$, tem-se que $\mathbf{S} = \mathbf{A}$. Por outro lado, quando $\lambda = 1$, $\mathbf{S} = \mathbf{I}$.

Considere agora as ELMs, apresentadas na Seção II-A. É importante ressaltar que outro método de classificação supervisionada poderia ser utilizado mas, devido à sua rapidez e à sua simplicidade, as ELM estão sendo tratadas neste trabalho. Considere, ainda, um problema de classificação semissupervisionada binária, no qual o conjunto rotulado $\mathbf{D}_L = \{\mathbf{x}_i, y_i\}_{i=1}^{N_L}$ é tal que $y_i \in \{-1, +1\}$. Para o conjunto de dados não-rotulados $\mathbf{D}_U$, são definidos pseudo-rótulos, tais que $\mathbf{D}_U = \{\mathbf{x}_j, y_j\}_{j=1}^{N_U}$ e $y_j = 0$. Assim, o conjunto $\mathbf{Y}^*$ de rótulos é tal que $\mathbf{Y}^* = \mathbf{Y} \cup \mathbf{Y}_0$, onde $\mathbf{Y}_0$ é o conjunto de pseudo-rótulos, todos iguais a 0.

A matriz de afinidade $\mathbf{A}$, na sua forma ponderada $\mathbf{S}$, será utilizada de forma a permitir que a ELM possa realizar classificação semissupervisionada, de três formas:

- **Configuração 1 ($\mathbf{H}'$)**: na primeira forma, é definida a matriz $\mathbf{H}'$, por meio da multiplicação da matriz $\mathbf{S}$ pela matriz $\mathbf{H}$ (Equação 1 - a saída da camada escondida da ELM):

$$\mathbf{H}' = \mathbf{S}\mathbf{H}. \tag{12}$$

Considerando $\lambda = 0$, tem-se que $\mathbf{H}'$ é a matriz $\mathbf{H}$ alterada de forma que cada valor h'_{kj} da mesma é

ponderado pelos respectivos valores a_{ik} de afinidade da matriz $\mathbf{A}$:

$$h'_{ij} = \Psi(\sum_{k=1}^{N}(a_{ik})h_{kj}), \tag{13}$$

onde $\Psi(\cdot)$ é uma função de ativação que pode ou não ser aplicada. Essa ponderação significa que a projeção de cada padrão na camada escondida é ponderada pelos valores de afinidade entre esse padrão e os demais.

- **Configuração 2 ($\mathbf{Y}'$)**: na segunda forma, define-se a matriz $\mathbf{Y}'$, multiplicando-se $\mathbf{S}$ pela matriz $\mathbf{Y}^*$:

$$\mathbf{Y}' = \mathbf{S}\mathbf{Y}^*. \tag{14}$$

Novamente considerando $\lambda = 0$, a definição de $\mathbf{Y}'$ significa que cada y_i é ponderado pela afinidade do padrão i com os demais padrões:

$$y'_i = \Psi(\sum_{k=1}^{N}(a_{ik})y_k), \tag{15}$$

onde $\Psi(\cdot)$ é uma função de ativação que pode ou não ser aplicada.

- **Configuração 3 ($\mathbf{H}'\mathbf{Y}'$)**: a terceira forma é a composição das duas formas anteriores, ou seja, a multiplicação por $\mathbf{S}$ é feita tanto em $\mathbf{H}$ como em $\mathbf{Y}^*$.

Quando é utilizada a matriz $\mathbf{Y}'$, isto é, quando $\mathbf{Y}^*$ é multiplicada por $\mathbf{S}$ (segunda e terceira formas acima), um novo passo é necessário: os rótulos originais são atribuídos a $\mathbf{Y}'$ nas posições de origem, isto é:

$$y'_i \leftarrow y^*_i, \ se \ y^*_i \neq 0. \tag{16}$$

Isso é feito para manter a informação fornecida pelo conjunto rotulado inalterada.

Assim, na ELM, a Equação 2 é alterada de acordo com cada configuração:

1) *Configuração 1*:

$$\mathbf{W} = \mathbf{H}'^{\dagger}\mathbf{Y}. \tag{17}$$

2) *Configuração 2*:

$$\mathbf{W} = \mathbf{H}^{\dagger}\mathbf{Y}'. \tag{18}$$

3) *Configuração 3*:

$$\mathbf{W} = \mathbf{H}'^{\dagger}\mathbf{Y}'. \tag{19}$$

O restante do treinamento da ELM segue inalterado. Ao fim do mesmo, é obtido o conjunto $\hat{\mathbf{Y}}$ de rótulos para os dados originalmente não-rotulados.

IV. Experimentos e Resultados

O procedimento para a realização dos experimentos foi o seguinte:

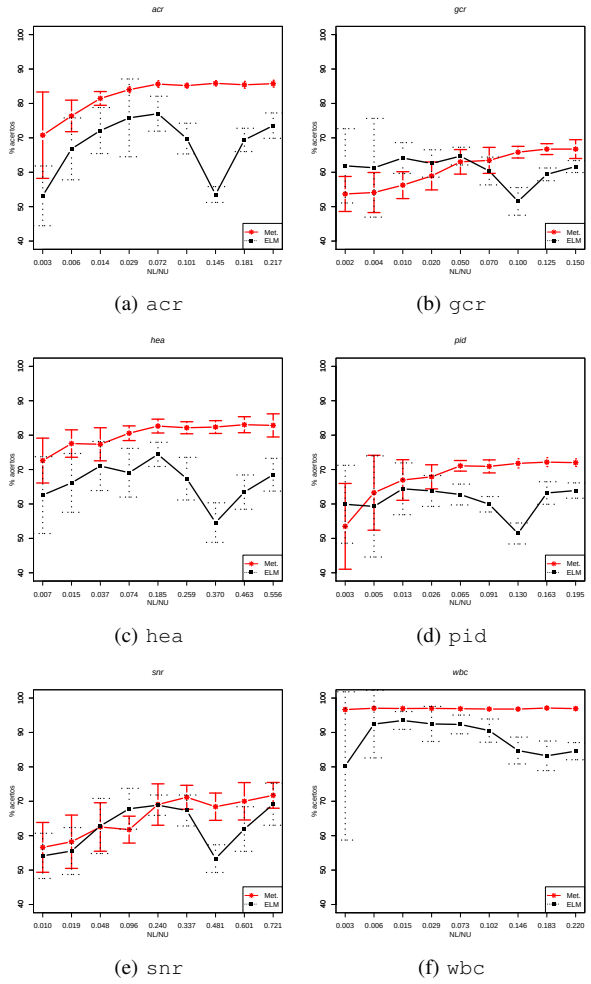
- Remoção dos padrões com valores ausentes; padronização dos rótulos para $\{-1, +1\}$; normalização (média 0 e desvio-padrão 1);
- Definição de: número de neurônios na camada escondida da ELM; configuração a ser utilizada para o método: $\mathbf{H}'$, $\mathbf{Y}'$ ou $\mathbf{H}'\mathbf{Y}'$; valor de λ (definido como 0);
- Criação de um vetor que determina a quantidade de dados rotulados: $NL = [2, 4, 10, 20, 50, 70, 100, 125, 150]$;
- Para cada valor nl_i de NL, repita (configurado para 10 repetições):
  - Seleção aleatória dos padrões que terão o rótulo mantido, de acordo com o valor definido em NL (nl_i), e armazenamento dos rótulos restantes para verificação posterior;
  - Geração de $\mathbf{Y}^*$: atribuição de 0 para os rótulos que não serão mantidos;
  - Cálculo da matriz de afinidade $\mathbf{A}$, de acordo com o método escolhido (FPM ou *Kernel* Linear);
  - Projeção dos dados no espaço de caraterísticas ELM (Equação 1);
  - De acordo com a versão selecionada, cálculo de $\mathbf{H}'$, $\mathbf{Y}'$ ou $\mathbf{H}'$ e $\mathbf{Y}'$;
  - Cálculo de $\mathbf{W}$ (Equação 2);
  - Projeção dos dados não-rotulados (Equação 6);
  - Cálculo da saída da rede, $\hat{\mathbf{Y}}_{test}$ (Equação 5);
  - Cálculo da acurácia, baseada no acerto de classificação dos dados selecionados para serem não-rotulados;
  - Treinamento da ELM original com o conjunto de dados rotulados definido;
  - Cálculo da acurácia de teste da ELM original, baseada no acerto de classificação dos dados selecionados para serem não-rotulados;
- Cálculo da média e do desvio-padrão dos resultados obtidos.

É importante observar que, para o cálculo de $\mathbf{Y}'$, quando utilizado *Kernel* Linear como matriz de afinidade, foi aplicada a função `sinal` como função de ativação. Já para o caso de usar FCM como matriz de afinidade, foi necessário encontrar o limiar para o cálculo de $\mathbf{Y}'$, pois o simples cálculo da função `sinal` como função de ativação não foi suficiente.

As bases de dados foram obtidas no repositório da UCI (*University of California, Irvine*) [16]. Foram utilizadas as bases *Australian Credit* (`acr`), *German Credit* (`gcr`), *Statlog (Heart)* (`hea`), *Pima Indians Diabetes* (`pid`), *Sonar, Mines vs. Rocks* (`snr`) e *Wisconsin Breast Cancer* (`wbc`).

A Figura 2 apresenta os gráficos dos resultados de classificação do método proposto e da ELM, para as bases reais, com o método configurado para utilizar *Kernel* Linear como matriz de afinidade, versão $\mathbf{Y}'$ e $p = 100$ neurônios na

camada escondida (eixo x: relação entre o número de padrões rotulados e o número de padrões não-rotulados - NL/NU; eixo y: porcentagem de acertos de classificação).



(a) `acr` (b) `gcr`

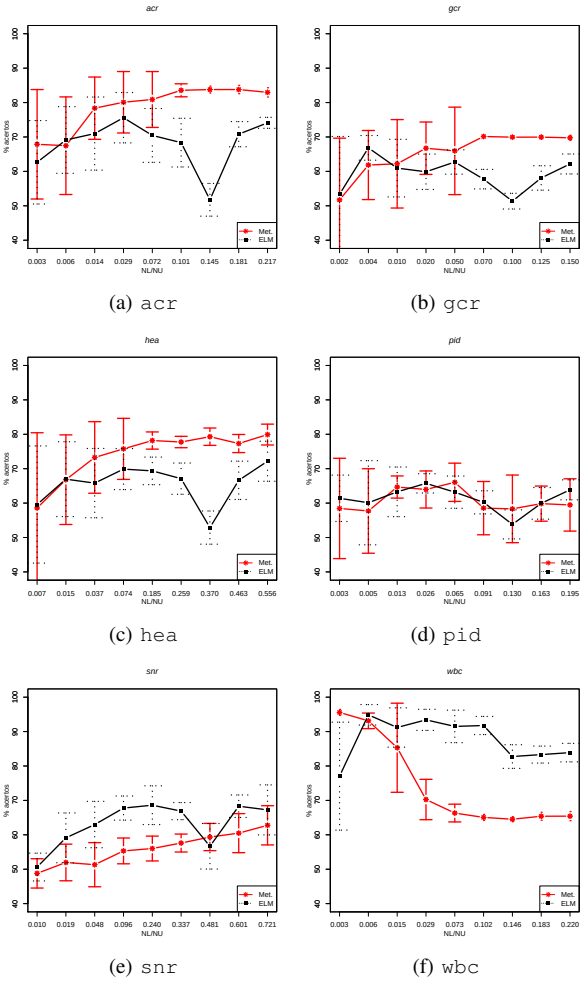
(c) `hea` (d) `pid`

(e) `snr` (f) `wbc`

Figura 2: Resultados de classificação - configuração: *Kernel* Linear, $p = 100$ e $\mathbf{Y}'$ (eixo x: relação entre o número de padrões rotulados e o número de padrões não-rotulados - NL/NU; eixo y: % de acertos de classificação).

A Figura 3 apresenta os gráficos dos resultados de classificação do método proposto e da ELM, para as bases reais, com o método configurado para utilizar FPM como matriz de afinidade, versão $\mathbf{Y}'$ e $p = 100$ neurônios na camada escondida (eixo x: relação entre o número de padrões rotulados e o número de padrões não-rotulados - NL/NU; eixo y: porcentagem de acertos de classificação).

No que diz respeito aos resultados de classificação, ainda para a versão que utiliza *Kernel* Linear, as médias dos resultados obtidos são melhores que os da ELM para as bases `acr`, `hea`, `pid` (exceto para $NL = 2$) e `wbc`. Para a base `snr`, o método apresenta resultados similares aos da ELM, obtendo ligeira vantagem quando $NL = [2, 4, 70]$ e grande vantagem quando $NL = 100$, isto é, $NL = p$ - o que é uma limitação conhecida da ELM, pois, nessa situação, a ELM aprende exatamente os padrões de treinamento, perdendo capacidade de generalização. Para a base `gcr`, o método apresenta vantagem



(a) `acr` (b) `gcr`

(c) `hea` (d) `pid`

(e) `snr` (f) `wbc`

Figura 3: Resultados de classificação - configuração: FPM, $p = 100$ e $\mathbf{Y}'$ (eixo x: relação entre o número de padrões rotulados e o número de padrões não-rotulados - NL/NU; eixo y: % de acertos de classificação).

apenas quando NL está próximo de p.

A versão que utiliza FCM como matriz de proximidade apresentou, excetuando-se alguns poucos valores de NL, melhores resultados que a ELM nas bases `acr`, `gcr` e `hea`. Para as demais bases, a ELM obteve vantagem na grande maioria dos valores de NL.

V. Discussões e Conclusões

Os resultados obtidos, apresentados na Seção IV, indicam a validade do método. Quando utilizado *Kernel* Linear como matriz de afinidade, o método apresentou melhores resultados do que a ELM para a maioria das bases. Por outro lado, os resultados obtidos pelo método na configuração que utiliza a FPM como matriz de afinidade foram piores do que aqueles obtidos pela ELM para a maioria das bases. Uma causa da vantagem do *Kernel* Linear - o qual também incorpora informações sobre os agrupamentos-, pode estar nas magnitudes dos valores da matriz, que são significativamente maiores do que as da FCM. Outra possível causa para esses

piores resultados da versão com FCM pode estar na forma que é gerada $\mathbf{Y}'$, pois é feita a ponderação por todos os valores rotulados, o que pode estar prejudicando a obtenção do rótulo correto, já que o valor sofre interferência dos valores de afinidade de todos os padrões rotulados. Uma abordagem que parece ser mais adequada é a de controlar a ponderação realizada para gerar $\mathbf{Y}'$, de forma que apenas um certo número de padrões rotulados que possuem maior afinidade com o padrão em questão interfiram na geração do rótulo do mesmo: assim, esse padrão terá uma probabilidade maior de receber o mesmo rótulo dos padrões mais afins.

O uso do *Kernel* Linear, além de possibilitar melhores resultados de classificação, possui a vantagem de não possuir parâmetros para serem ajustados, além de o cálculo do produto interno ser simples e rápido. Por sua vez, além da desvantagem nos resultados, o cálculo da FPM envolve o uso de um algoritmo de *clustering*, o qual depende de um parâmetro: o número de agrupamentos. Além disso, encontrar a FPM demanda um custo computacional maior do que calcular o *Kernel* Linear. Essas constatações apontam para investigações acerca do uso de outros tipos de *kernel*, como o Gaussiano. A investigação deve considerar se o ganho será significativo a ponto de justificar o ajuste do parâmetro do *Kernel* Gaussiano: o raio RBF.

AGRADECIMENTOS

O presente trabalho foi realizado com o apoio financeiro da CAPES - Brasil, da Fapemig - Fundação de Amparo à Pesquisa do estado de Minas Gerais e do CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico, aos quais os autores agradecem.

REFERÊNCIAS

[1] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*, 2nd ed. Wiley-Interscience, November 2000.

[2] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 2, july 2004, pp. 985–990 vol.2.

[3] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, December 2006. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2005.12.126

[4] D. Serre, *Matrices: Theory and Applications*, 1st ed. Springer, November 2002.

[5] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, vol. 14, pp. 326–334, 1965.

[6] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York: Macmillan, 1994.

[7] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006. [Online]. Available: http://www.kyb.tuebingen.mpg.de/ssl-book

[8] J. Liu, Y. Chen, M. Liu, and Z. Zhao, "Selm: Semi-supervised elm with application in sparse calibrated location estimation," *Neurocomputing*, vol. 74, no. 16, pp. 2566–2572, 2011.

[9] M. Belkin, I. Matveeva, and P. Niyogi, "Regularization and semi-supervised learning on large graphs," in *COLT*, 2004, pp. 624–638.

[10] K. Li, J. Zhang, H. Xu, S. Luo, and H. Li, "A semi-supervised extreme learning machine method based on co-training," *Journal of Computational Information Systems*, vol. 9, pp. 207–214, 2013.

[11] A. d. P. Braga, "Notas de aula - redes neurais artificiais," 2012, notas de aula da disciplina Redes Neurais Artificiais - Programa de Pós Graduação em Engenharia Elétrica - Universidade Federal de Minas Gerais.

[12] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, 1995, pp. 273–297.

[13] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 1981.

[14] F. Queiroz, A. Braga, and W. Pedrycz, "Sorted Kernel Matrices as Cluster Validity Indexes," in *European Society for Fuzzy Logic and Technology*, 2009, pp. 1490–1495.

[15] F. A. de Andrade Queiroz, "Um estudo sobre métodos de kernel para classificação e agrupamento de dados," Master's thesis, Universidade Federal de Minas Gerais, 2009.

[16] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/mlearn/MLRepository.html