LEONARDO JOSÉ SILVESTRE

REGULARIZAÇÃO DE EXTREME LEARNING MACHINES: UMA ABORDAGEM COM MATRIZES DE AFINIDADE

REGULARIZAÇÃO DE EXTREME LEARNING MACHINES: UMA ABORDAGEM COM MATRIZES DE AFINIDADE

LEONARDO JOSÉ SILVESTRE

Tese de doutorado submetida à banca examinadora designada pelo colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Prof. Dr. Antônio de Pádua Braga
Co-orientador: Prof. Dr. André Paim Lemos

Belo Horizonte - MG
Fevereiro de 2015

Dedico este trabalho à memória do meu irmão Leandro.

Dedico, ainda, à minha esposa Rúbia e aos meus filhos Maria Eduarda e Davi.

RESUMO

O problema de indução de modelos a partir de um conjunto de dados é um problema inverso e, tipicamente, mal condicionado. Para tornar esse problema bem condicionado, técnicas de regularização têm sido utilizadas com sucesso, inclusive nas Redes Neurais Artificiais (RNAs). Tais técnicas utilizam informação obtida *a priori* sobre o problema, a qual pode ser, por exemplo, imposição de suavidade da solução, ou ainda, informação estrutural sobre os dados a serem tratados. Essa informação estrutural sobre os dados pode ser provida por matrizes de afinidade - por exemplo, matrizes de *Kernel* e matrizes de similaridade de cossenos. No contexto das RNAs, um algoritmo para treinamento de Redes Neurais de uma única Camada Alimentadas Adiante - *Single-Layer Feedforward Neural Networks* (SLFNs), chamado de Máquina de Aprendizado Extremo - *Extreme Learning Machine* (ELM), tem recebido atenção da comunidade científica nos últimos anos, especialmente por causa da sua simplicidade e rapidez de treinamento. Por ter seu treinamento feito em duas etapas: projeção aleatória em um espaço de alta dimensão e cálculo dos pesos da camada de saída por meio da pseudo inversa, o algoritmo da rede ELM permite intervenções de forma a inserir informações obtidas por meio de matrizes de afinidade em seu treinamento. Isso pode ser feito combinando as projeções obtidas pela rede ELM com as matrizes de afinidade. Neste trabalho, é demonstrado que o uso desse tipo de informação estrutural no treinamento das ELMs possibilita um efeito similar à regularização de Tikhonov. Além disso, essa modificação no algoritmo da ELM possibilita que a mesma possa ser utilizada no contexto do aprendizado semissupervisionado. Nesse tipo de aprendizado, no qual os rótulos são escassos, em geral utiliza-se informação estrutural dos dados para auxiliar na construção do modelo. Experimentos realizados com o algoritmo desenvolvido, chamado de ELM Regularizada com Matrizes de Afinidade - *Affinity Matrix Regularized ELM* (AMR-ELM), mostram a validade do método, validando tanto o efeito de regularização obtido no contexto do aprendizado supervisionado quando a capacidade de lidar com a escassez de rótulos própria do aprendizado semissupervisionado. Além disso, o uso de uma matriz de afinidade sem parâmetros, tal como a matriz de similaridade de cossenos, possibilita que a regularização não necessite de ajuste de parâmetros.

ABSTRACT

Inducing models from a dataset is an inverse problem and usually it is ill-posed. To turn this into a well-posed problem, regularization techniques have been used with success, including in Artificial Neural Networks (ANN). These techniques use *a priori* information about the problem. This information may be, for example, imposing smoothness to the solution, or using structural information about the dataset. The structural information can be provided by affinity matrices - for example, kernel matrices and cosine similarity matrices. In the ANN context, a Single-Layer Feedforward Neural Network (SLFN) training algorithm has been attracting attention of the scientific community in recent years, especially because of its simplicity and speed of training. Because its training is done in two steps: random projection in a high-dimensional space and calculating the output layer weights using the pseudo-inverse, the ELM algorithm allows to interfere on it in order to insert information obtained by affinity matrices. This can be done by combining the ELM projections with the affinity matrices. In this thesis, we show that using such structural information in ELM training provides an effect similar to Tikhonov regularization. Moreover, this change in ELM algorithm enables it to be used in the semi-supervised learning context. This type of learning, in which labels are scarce, usually uses structural information about the data in order to help model construction. Experiments performed with the developed algorithm, which we call Affinity Matrix Regularized ELM (AMR-ELM), validate both the regularization effect in the context of supervised learning and the ability to deal with semi-supervised learning scarcity of labels. Furthermore, if a parameter-free affinity matrix is used, like the cosine similarity matrix, regularization is performed without any need for parameter tunning.

PUBLICAÇÕES

Durante o desenvolvimento deste trabalho, os seguintes trabalhos foram publicados:

SILVESTRE, L. J. ; LEMOS, A. P. ; BRAGA, J. P. ; Braga, A. P. **Dataset Structure as Prior Information for Parameter-Free Regularization of Extreme Learning Machines.** In: *Neurocomputing (aceito para publicação).*

SILVESTRE, L. J. ; LEMOS, A. P. ; BRAGA, J. P. ; Braga, A. P. **Parameter-free regularization in Extreme Learning Machines with affinity matrices.** In: *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, ESANN 2014, 23-25 April 2014, Bruges, Belgium. Louvain-la-Nueve, Belgique, 2014. p. 595-600.

SILVESTRE, L. J. ; Braga, A. P. **Aprendizado Semissupervisionado com Extreme Learning Machines e Matrizes de Afinidade**. In: 11th Brazilian Congress (CBIC) on Computational Intelligence, 2013, Ipojuca - PE. *Proceedings of 1st BRICS Countries Congress (BRICS-CCI) and 11th Brazilian Congress (CBIC) on Computational Intelligence*, 2013.

SILVESTRE, L. J.; CASTRO, C. L.; Braga, A. P. **Treinamento de redes perceptron de multiplas a camadas utilizando o conceito de sparse distributed memory**. *In Anais do XVIII Congresso Brasileiro de Automática*, CBA, pages 3886-3891, 2010.

AGRADECIMENTOS

- A Deus, por tudo o que sou e tenho, e pela força em todos os momentos, especialmente nos mais difíceis.
- A minha esposa Rúbia, pelo amor, pelo carinho, pelo cuidado e pela paciência, em especial nos momentos de ausência necessários para dedicar-me a este trabalho.
- Aos meus filhos Maria Eduarda e Davi, por todas as alegrias, pelos carinhos e pelos sorrisos, "disponibilizados à vontade".
- A meus pais, José Amorim Silvestre e Maria de Lourdes Silvestre, assim como aos demais familiares, pela educação recebida, pelos exemplos, pelo apoio e pelas orações.
- A meu orientador, Professor Antônio de Pádua Braga, por guiar-me neste caminho até aqui com tanta dedicação e boa vontade. Sem sua orientação, certamente eu não teria chegado perto de onde cheguei. Registro aqui a minha admiração por sua pessoa, e certamente as palavras serão poucas para agradecê-lo.
- A meu co-orientador, Professor André Paim Lemos, pela fundamental contribuição a este trabalho, especialmente na formalização do método desenvolvido.
- Aos demais professores do PPGEE, pelo conhecimento recebido.
- Aos amigos e colegas do LITC, pelos momentos de convivência, pelas discussões e pelos auxílios.
- Aos amigos e colegas do DCEL/CEUNES/UFES e do DCOMP/UFSJ, pelo apoio e por "segurarem as pontas" nos momentos de minha ausência.
- A todos os demais amigos, pelo apoio e pela torcida.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIAÇÕES

AMR-ELM ELM Regularizada com Matrizes de Afinidade - *Affinity Matrix Regularized ELM*

BIC	Critério de Informação Bayesiana - *Bayesian Information Criterion*
CDD	Diagrama de Diferença Crítica - *Critical Difference Diagram*
DE	Evolução Diferencial - *Differential Evolution*
DG	*Data Generator*
E-ELM	ELM Evolucionária - *Evolutionary ELM*
EBP	Retropropagação de Erros - *Error Back Propagation*
ELM	Máquina de Aprendizado Extremo - *Extreme Learning Machine*
EOS-ELM	*Ensemble of Online Sequential Extreme Learning Machines*
i.i.d.	independente e identicamente distribuídos
I-ELM	ELM Incremental - *Incremental Extreme Learning Machine*
LOO	*Leave-One-Out*
LS-SVM	SVM de Mínimos Quadrados - *Least Squares Support Vector Machine*
MLP	Perceptron de Múltiplas Camadas - *Multi-Layer Perceptron*
OLS	Mínimos Quadrados Ordinários - *Ordinary Least Squares*
OP-ELM	ELM Podada de forma Ótima - *Optimally Pruned Extreme Learning Machine*
OS-ELM	ELM Sequencial Online - *Online Sequential Extreme Learning Machine*
PSVM	*Proximal Support Vector Machine*
RBF	Função de Base Radial - *Radial Basis Function*
RMSE	Raiz do Erro Quadrático Médio - *Root Mean Squared Error*
RNA	Rede Neural Artificial
SELM	Máquina de Aprendizado Extremo Semissupervisionada - *Semi-Supervised ELM*

SLFN	Rede Neural de uma única Camada Alimentada Adiante - *Single-Layered Feedforward Neural Network*
SVM	Máquina de Vetores Suporte - *Support Vector Machine*
SVD	Decomposição em Valores Singulares - *Singular Value Decomposition*
SVR	Regressão por Vetores Suporte - *Support Vector Regression*
TER-ELM	ELM de Taxa de Erro Total - *Total Error Rate ELM*
TROP-ELM	ELM Podada de forma Ótima com Regularização de Tikhonov - *Tikhonov Regularized Optimally Pruned ELM*

1 INTRODUÇÃO

Modelos de aprendizado supervisionado são induzidos a partir de um conjunto de dados $\mathbf{D}$ composto por N pares de padrões ($\mathbf{x}$) e rótulos (y): $\mathbf{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$. O problema de indução de modelos a partir de dados é um problema inverso e, com frequência, mal condicionado. Com o objetivo de tornar esse problema bem condicionado, técnicas de regularização, introduzidas por Tikhonov (Tikhonov, 1963), têm sido utilizadas com sucesso, inclusive nas RNAs (Girosi et al., 1995; Chen e Haykin, 2002). Usualmente, as técnicas de regularização utilizam alguma informação, obtida *a priori*, sobre o problema. Essa informação pode ser dada, por exemplo, por uma imposição de suavidade na solução. Informação *a priori* para regularização também pode ser obtida por meio da informação estrutural dos dados a serem tratados no problema.

No contexto das RNAs, uma área tem atraído a atenção da comunidade acadêmica: as Máquinas de Aprendizado Extremo - *Extreme Learning Machines* (ELMs) (Huang et al., 2004, 2006a). Diferentemente de outros algoritmos para treinamento de RNAs, que fazem o ajuste dos parâmetros da rede por meio da apresentação iterativa dos padrões de treinamento, as ELMs fazem uma projeção aleatória na camada escondida, a qual possui, em geral, alta dimensionalidade. Os pesos dessa camada são selecionados de forma aleatória, sem necessidade de treinamento e, após a projeção, os pesos da camada de saída são calculados de forma analítica, utilizando a inversa generalizada de Moore-Penrose, ou pseudo inversa (Serre, 2002). Assim, o treinamento das ELMs é feito de forma rápida e simples, e a rede resultante possui capacidade de generalização comparável a de outras Redes Neurais de uma única Camada Alimentadas Adiante - *Single-Layer Feedforward Neural Networks* (SLFNs). Além disso, as ELMs são simples de configurar, já que, geralmente, o único parâmetro a ser ajustado é o número de neurônios na camada escondida.

Quando comparadas com RNAs treinadas com algoritmos tradicionais, como o Retropropagação de Erros - *Error Back Propagation* (EBP) (Rumelhart et al., 1986), especialmente no contexto da classificação de padrões, a ELM é comumente sobredimensionada. Isso decorre do fato de não haver, na ELM, ajuste dos pesos da camada escondida, sendo necessário projetar os padrões em um espaço de dimensão mais alta: segundo o Teorema de Cover (Cover, 1965), isso é necessário para se obter uma maior probabilidade de separabilidade linear. Dado esse sobredimensionamento da ELM, alguma suavização da resposta é necessária para garantir a qualidade da sua capacidade de generalização. Nesse

sentido, modificações na ELM vêm sendo propostas para incluir um termo de regularização durante a estimação dos pesos da camada de saída (Deng et al., 2009; Miche et al., 2011; Yu et al., 2013).

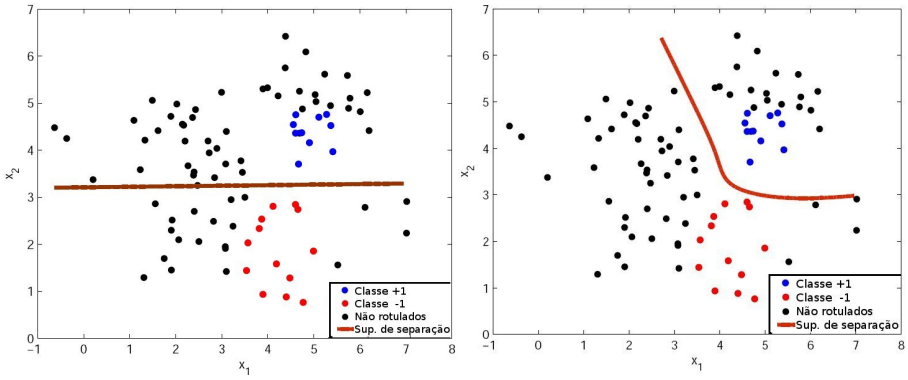
Alternativamente, podem ser usadas informações sobre a estrutura dos dados de forma a obter a regularização do problema. Uma matriz de afinidade (ou matriz de similaridade) é uma matriz que contém medidas de similaridade entre cada par de observações de um determinado conjunto de dados. Essas matrizes, que têm sido utilizadas em diversos problemas de análise de dados, incluindo agrupamento (*clustering*), redução de dimensionalidade, segmentação de imagens e análise de relacionamentos entre objetos (*link analysis*) (Rosales e Frey, 2002), contêm informação importante sobre a estrutura dos dados. Por isso, elas aparentam ser boas candidatas para auxiliar no processo de regularização, por meio de uso da informação *a priori* sobre a estrutura dos dados.

Como o treinamento das ELMs é realizado em duas etapas, é possível visualizar uma forma de inserir informações estruturais dos dados, fornecidas por matrizes de afinidade, para obter o efeito de regularização, melhorando a capacidade de generalização das mesmas.

Para possibilitar a indução de modelos, no aprendizado supervisionado, para cada padrão $\mathbf{x}_i$ deve existir um rótulo y_i. Entretanto, uma situação muito comum é haver uma quantidade grande de padrões $\mathbf{x}_i$ para os quais não há rótulos disponíveis. Essa diferença de disponibilidade de padrões rotulados e não-rotulados decorre do fato de que, apesar de a quantidade de dados disponíveis ser, em geral, muito grande, determinar os rótulos pode ser extremamente custoso, já que esse processo envolve a participação de especialistas no domínio, equipamentos especiais ou, ainda, experimentos caros e demorados (Zhu et al., 2009). Por exemplo, gravar a fala é um processo que tem custo relativamente baixo, mas rotular corretamente a fala gravada pode ser custoso (Duda et al., 2000). Algoritmos de aprendizado semissupervisionado lidam com essa situação, na qual o conjunto de dados $\mathbf{D}$ é tal que $\mathbf{D}_L \cup \mathbf{D}_U$, onde $\mathbf{D}_L = \{\mathbf{x}_i, y_i\}_{i=1}^{N_L}$ - dados rotulados - e $\mathbf{D}_U = \{\mathbf{x}_j\}_{j=1}^{N_U}$ - dados não-rotulados, e $N_U \gg N_L$, com N_L sendo o número de dados rotulados e N_U o número de dados não-rotulados.

Para suprir a ausência de rótulos, o aprendizado semissupervisionado pode utilizar informação sobre a estrutura dos dados. Por exemplo, considere a Figura 1.1, que apresenta um problema de classificação binária e duas soluções distintas para esse problema. Os padrões rotulados de uma classe são apresentados em azul, e os da outra classe, em vermelho. Os padrões de cor preta não possuem rótulos. As linhas avermelhadas apresentam as superfícies de decisão (ou separação) obtidas por meio de aprendizado supervisionado e semissupervisionado: no primeiro caso, representado pela Figura 1.1a, o aprendizado supervisionado leva em consideração apenas os dados rotulados, resultando em uma superfície de separação bastante diferente daquela apresentada na Figura 1.1b,

resultante de aprendizado semissupervisionado. Este levou em consideração também a estrutura apresentada pelos dados não-rotulados, realizando a separação baseando-se no princípio de que esta deve ocorrer em uma região de baixa densidade. Ou seja, ao desconsiderar a grande quantidade de dados não-rotulados, o algoritmo de aprendizado supervisionado perde parte importante da informação sobre a estrutura dos dados, apresentando um resultado de classificação que pode não ser o mais adequado.



(a) (b)

Figura 1.1: Superfícies de decisão geradas a partir dos dados rotulados (a) e dos dados rotulados e não-rotulados (b) (Braga, 2012).

Assim, caso seja possível utilizar matrizes de afinidade nas ELMs, inserindo informações estruturais dos dados em seu treinamento, as mesmas podem tornar-se capazes de lidar também com aprendizado semissupervisionado.

1.1 MOTIVAÇÕES E OBJETIVOS DO TRABALHO

Tanto a regularização de modelos de aprendizado supervisionado quanto o aprendizado semissupervisionado podem utilizar informação obtida diretamente da estrutura dos dados: a primeira para possibilitar a regularização em si, a segunda para induzir o modelo com um conjunto de dados rotulados escasso. Entre as diversas formas de extrair informação dos dados encontram-se as matrizes de afinidade (ou similaridade), as quais são matrizes tais que cada elemento contém uma medida de afinidade entre dois padrões. Assim, as matrizes de afinidade expressam as relações entre as amostras e entre grupos de amostras presentes nos conjuntos de dados - informação estrutural que pode ser útil na construção de modelos. Matrizes de *Kernel* (Haykin, 1994), produto interno de matrizes e matriz de similaridade de cossenos são exemplos importantes de matrizes de afinidade.

O fato de o treinamento das ELMs ser feito em duas etapas facilita o desenvolvimento de técnicas que interfiram em alguma das etapas para adicionar informações de matrizes de afinidade. A ideia está baseada

em combinar as projeções obtidas pela ELM com as matrizes de afinidade, as quais contêm informações estruturais dos dados. Dessa forma, no caso da regularização de modelos supervisionados, a informação *a priori* fornecida por essas matrizes provê o efeito de regularização. Já no aprendizado semissupervisionado, o modelo resultante é induzido não somente por meio do modelo obtido a partir do conjunto rotulado, mas também por meio das relações de afinidade entre elementos dos conjuntos de dados rotulados e não-rotulados.

Vale destacar que, dependendo do tipo de afinidade utilizada, a regularização pode ser obtida sem a necessidade de se ajustar parâmetros, o que representa um ganho significativo no tempo de construção de modelos regularizados.

1.1.1 *Objetivos e Contribuições*

Este trabalho de doutorado tem como objetivo geral desenvolver um método para utilizar informação da estrutura dos dados, proveniente de matrizes de afinidade, para aprimorar o desempenho de classificadores. Os objetivos específicos para alcançar esse objetivo geral são:

- Conhecer e estudar algoritmos de aprendizado supervisionado para classificação, em especial as *Extreme Learning Machines*;
- Conhecer e estudar estratégias de extração de informação de estrutura dos dados, especialmente matrizes de afinidade;
- Conhecer e estudar a regularização de problemas mal condicionados, especialmente a regularização em aprendizado de máquina;
- Conceber e desenvolver um método de regularização, baseado em matrizes de afinidade, para a ELM;
- Conceber e desenvolver um método de classificação semissupervisionada, baseado em matrizes de afinidade, para a ELM;
- Avaliar os métodos desenvolvidos em problemas reais, comparando os resultados obtidos com métodos similares na literatura.

Entre as contribuições, pode-se citar:

- Desenvolvimento formal demonstrando que o uso de matrizes de afinidade normalizadas no treinamento de ELMs leva a um efeito de regularização de Tikhonov (Tikhonov, 1963) e, ainda, permite estimar rótulos quando parte destes está ausente, como no caso do aprendizado semissupervisionado;
- Proposta de um algoritmo que implementa esse desenvolvimento, o qual pode ser aplicado tanto a problemas de aprendizado supervisionado quanto a problemas de aprendizado semissupervisionado.

Os resultados preliminares, tratando apenas do aprendizado semissupervisionado, encontram-se publicados em (Silvestre e Braga, 2014), enquanto os resultados provenientes da formalização do efeito de regularização foram publicados em (Silvestre et al., 2014). Este último trabalho recebeu um convite para integrar a edição especial do congresso em uma revista internacional. Tal extensão foi submetida e aceita para publicação.

1.2 ORGANIZAÇÃO DO DOCUMENTO

O restante deste texto está organizado da seguinte forma:

O Capítulo 2 apresenta a revisão bibliográfica, abordando Regularização, *Extreme Learning Machines*, Aprendizado Semissupervisionado e Matrizes de Afinidade. O Capítulo 3 apresenta o método para regularização em ELMs desenvolvido, chamado de AMR-ELM. O Capítulo 4 apresenta os experimentos realizados, os resultados encontrados e as discussões sobre esses resultados e, por fim, no Capítulo 5, são apresentadas as conclusões e os trabalhos futuros.

2

REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta a revisão bibliográfica sobre os principais assuntos tratados neste texto: Regularização, Máquinas de Aprendizado Extremo - *Extreme Learning Machines*, Aprendizado Semissupervisionado e Matrizes de Afinidade. Antes de apresentar esses assuntos, são apresentadas as definições utilizadas, para melhor entendimento.

2.1 DEFINIÇÕES

Considere o conjunto de dados $D = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$, composto de N pares de vetores $(\mathbf{x}_i, \mathbf{y}_i)$, em que $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_N]^T e \mathbf{Y} = [\mathbf{y}_1\ \mathbf{y}_2\ \ldots\ \mathbf{y}_N]^T$, conforme representado nas Equações 2.1 e 2.2. As linhas de $\mathbf{X}$ contêm os N vetores de entrada de dimensão n, e as linhas de $\mathbf{Y}$ contêm as N respostas da rede neural para cada um dos M neurônios de saída. Portanto, a rede neural correspondente possui n entradas e m saídas.

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Nn} \end{bmatrix}_{N \times n}, \tag{2.1}$$

onde n é a dimensão do espaço de entrada e N o número de amostras.

$$\mathbf{Y} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N1} & y_{N2} & \cdots & y_{Nm} \end{bmatrix}_{N \times m}, \tag{2.2}$$

onde m é a dimensão do espaço de saída.

A matriz de pesos $\mathbf{Z}$ da camada escondida, com dimensões $n \times \ell$ é definida na Equação 2.3, e a matriz de pesos $\mathbf{W}$ da camada de saída, com dimensões $\ell \times m$ é definida na Equação 2.4. Assim, as colunas de $\mathbf{Z}$ contêm os vetores de pesos de cada um dos ℓ neurônios da camada

escondida. De forma análoga, as colunas de $\mathbf{W}$ contêm os vetores de pesos de cada um dos m neurônios de saída.

$$\mathbf{Z} = \begin{bmatrix} z_{11} & z_{12} & \dots & z_{1\ell} \\ z_{21} & z_{22} & \dots & z_{2\ell} \\ \vdots & \vdots & \ddots & \vdots \\ z_{n1} & z_{n2} & \dots & z_{n\ell} \end{bmatrix}_{n \times \ell}, \tag{2.3}$$

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{\ell 1} & w_{\ell 2} & \dots & w_{\ell m} \end{bmatrix}_{\ell \times m}, \tag{2.4}$$

O produto de $\mathbf{X}$ por $\mathbf{Z}$ resulta na matriz $\mathbf{U}$ de dimensão $N \times \ell$, que corresponde à resposta linear de todos os ℓ neurônios da camada escondida, conforme Equação 2.5.

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1\ell} \\ u_{21} & u_{22} & \dots & u_{2\ell} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1} & u_{N2} & \dots & u_{N\ell} \end{bmatrix}_{N \times \ell}, \tag{2.5}$$

Sobre a matriz $\mathbf{U}$ é aplicada a função de ativação comum a todos os neurônios da camada escondida, obtendo-se, dessa forma, a matriz $\mathbf{H}$, que contém os mapeamentos não-lineares de todas as amostras de entrada no espaço da camada escondida. A matriz $\mathbf{H}$, que tem a mesma dimensão de $\mathbf{U}$, ou seja, é uma matriz $N \times \ell$, pode ser representada pela função geral de mapeamento $h(\mathbf{X}, \mathbf{Z})$, e seus elementos pelas funções $h_i(\mathbf{x}_i, \mathbf{z}_i)$, conforme equação 2.6.

$$\mathbf{H} = \begin{bmatrix} h_1(\mathbf{x}_1, \mathbf{z}_1) & h_2(\mathbf{x}_1, \mathbf{z}_2) & \dots & h_\ell(\mathbf{x}_1, \mathbf{z}_\ell) \\ h_1(\mathbf{x}_2, \mathbf{z}_1) & h_2(\mathbf{x}_2, \mathbf{z}_2) & \dots & h_\ell(\mathbf{x}_2, \mathbf{z}_\ell) \\ \vdots & \vdots & \ddots & \vdots \\ h_1(\mathbf{x}_N, \mathbf{z}_1) & h_2(\mathbf{x}_N, \mathbf{z}_2) & \dots & h_\ell(\mathbf{x}_N, \mathbf{z}_\ell) \end{bmatrix}_{N \times \ell}, \tag{2.6}$$

onde ℓ corresponde à dimensão (número de neurônios) da camada intermediária.

A saída j do modelo responde com a aproximação $\hat{y}_{ij} = \phi_j(h(\mathbf{x}_i, \mathbf{Z}), \mathbf{w}_j)$ de y_{ij} para cada vetor de entrada $\mathbf{x}_i$, resultando na matriz de aproximação $\hat{\mathbf{Y}}$, apresentada na Equação 2.7, em que $\phi_j(\cdot, \cdot)$ é a função de ativação da saída j. O argumento $h(\mathbf{x}_i, \mathbf{Z})$ de $\phi_j(\cdot, \mathbf{w}_j)$ corresponde à

linha i da matriz de mapeamento $\mathbf{H}$. O elemento $\hat{y}_{ij}$ corresponde à saída j do modelo para o padrão de entrada $\mathbf{x}_i$, conforme Equação 2.7.

$$\hat{\mathbf{Y}} = \begin{bmatrix} \hat{y}_{11} & \hat{y}_{12} & \cdots & \hat{y}_{1m} \\ \hat{y}_{21} & \hat{y}_{22} & \cdots & \hat{y}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{y}_{N1} & \hat{y}_{N2} & \cdots & \hat{y}_{Nm} \end{bmatrix}_{N \times m} . \tag{2.7}$$

A matriz de erros de saída $\mathbf{E} = [e_{ij}]$ é obtida pelo desvio $\mathbf{E} = (\mathbf{Y} - \hat{\mathbf{Y}})$ entre a matriz de saída $\mathbf{Y}$ e a matriz correspondente $\hat{\mathbf{Y}}$ gerada pelo modelo, conforme apresentado na Equação 2.8.

$$\mathbf{E} = \begin{bmatrix} (y_{11} - \hat{y}_{11}) & (y_{12} - \hat{y}_{12}) & \cdots & (y_{1m} - \hat{y}_{1m}) \\ (y_{21} - \hat{y}_{21}) & (y_{22} - \hat{y}_{22}) & \cdots & (y_{2m} - \hat{y}_{2m}) \\ \vdots & \vdots & \ddots & \vdots \\ (y_{N1} - \hat{y}_{N1}) & (y_{N2} - \hat{y}_{N2}) & \cdots & (y_{Nm} - \hat{y}_{Nm}) \end{bmatrix}_{N \times m} . \tag{2.8}$$

2.2 REGULARIZAÇÃO

A maioria dos problemas do mundo real é mal condicionada (*ill-posed*). As condições para um problema ser bem-condicionado, conhecidas como condições de Hadamard, são (Hadamard, 1902; Velho, 2001):

- *Existência*: existe uma solução;
- *Unicidade*: a solução é única;
- *Continuidade ou estabilidade*: a solução possui uma dependência contínua com os dados de entrada, ou seja, pequenas mudanças nas condições iniciais e de contorno causam alterações pequenas na solução.

Caso qualquer uma dessas condições não seja satisfeita, o problema é considerado mal condicionado. Por exemplo, o problema da aproximação de funções a partir de um conjunto de dados é um problema mal condicionado: a informação contida no conjunto de amostras não é suficiente para a reconstrução única do mapeamento de entrada-saída em regiões nas quais não existem dados disponíveis (Girosi et al., 1995), ou seja, a solução não é única. A teoria da regularização foi desenvolvida inicialmente por Tikhonov (Tikhonov, 1963) para a solução de problemas mal condicionados de reconstrução de superfícies (Ferreira, 2005). Para tornar o problema bem condicionado, é necessário incluir conhecimento prévio sobre o mesmo. Na ausência de qualquer conhecimento, a única informação que pode ser inserida *a priori* diz respeito

ao elevado grau de suavidade da função a ser aproximada (Poggio e Girosi, 1990), ou seja, necessita-se previamente de que a solução seja suave, de forma a lidar com um conjunto de amostras insuficiente para reconstruir, de forma única, o mapeamento entrada-saída onde não há dados disponíveis (Poggio e Girosi, 1990).

No caso da regularização de Tikhonov, essa informação *a priori* é inserida por meio de uma penalidade ponderada (Tikhonov, 1963; Hoerl e Kennard, 1970). Por exemplo, considere o problema de Mínimos Quadrados Ordinários - *Ordinary Least Squares* (OLS):

$$\min_{x} \|\mathbf{A}\mathbf{x} - \mathbf{B}\|^2, \tag{2.9}$$

ou seja, a solução de mínimos quadrados para o sistema de equações $\mathbf{A}\mathbf{x} = \mathbf{B}$, no qual $\mathbf{A}$ é a matriz dos coeficientes, $\mathbf{x}$ é o vetor de incógnitas, e $\mathbf{B}$ é o vetor dos termos independentes. Normalmente, em problemas do mundo real, esse problema é mal condicionado. Adicionar um termo de regularização, $\|\mathbf{x}\|^2$, corresponde a inserir uma penalidade, a qual pode ser controlada por um parâmetro escalar λ. Para o problema de OLS, tem-se:

$$\min_{x} \|\mathbf{A}\mathbf{x} - \mathbf{B}\|^2 + \lambda\|\mathbf{x}\|^2. \tag{2.10}$$

O problema de OLS, assim, passa a ser o problema de Mínimos Quadrados Ordinários Regularizado (*Regularized* OLS). Quando o parâmetro λ é zero, tem-se o OLS original e, à medida que λ cresce, o termo de regularização torna-se mais importante. Assim, λ controla o compromisso (*trade-off*) entre a minimização do OLS original e a suavidade imposta por $\lambda\mathbf{x}$ (Scholkopf e Smola, 2001). A solução para esse novo problema é

$$\hat{\mathbf{x}} = (\mathbf{A}^\top\mathbf{A} + \Gamma^\top\Gamma)^{-1}\mathbf{A}^\top\mathbf{B}, \tag{2.11}$$

na qual $\Gamma = \sqrt{\lambda}\mathbf{I}$ e $\mathbf{I}$ é a matriz identidade. Assim, uma restrição de suavidade é imposta à solução. Um valor adequado de λ pode ser encontrado de várias formas, um exemplo seria via validação cruzada.

2.2.1 *Regularização e Aprendizado de Máquina*

O problema de aprendizado de máquina pode ser formalizado como um problema de aproximação de funções (Murphy, 2012), no qual deseja-se estimar uma função desconhecida $y = f(\mathbf{x})$ a partir de um conjunto de dados.

A Figura 2.1 apresenta um esquema geral para o aprendizado supervisionado. O gerador de dados *Data Generator* (DG) provê os dados $\mathbf{X}$ de acordo com a distribuição $P(\mathbf{X})$. O agente de rotulação, também

conhecido como Oráculo (O), o qual conhece uma aproximação para a distribuição condicional $P(\mathbf{Y}|\mathbf{X})$, onde $\mathbf{Y}$ é o conjunto de rótulos e $\mathbf{X}$ é o conjunto de padrões, provê um rótulo para cada dado amostrado pelo DG (o conjunto de dados de treinamento). Um modelo (M) pode ser ajustado com um padrão $\mathbf{x}$ e seu respectivo rótulo y, de forma a ser usado por um Estimador (E) para classificar ou prever o rótulo y^* para o novo padrão $\mathbf{x}^*$.

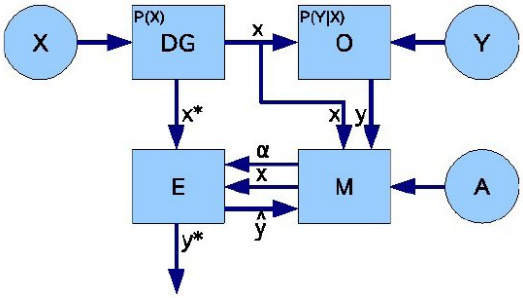


Figura 2.1: Esquema geral para o Aprendizado Supervisionado (Braga, 2012).

Em outras palavras, de um conjunto $\mathcal{D} = \{\mathbf{x}_i, y_i\}$, no qual cada $\mathbf{x}_i \in \mathbb{R}^n$ é um dado ou padrão e cada $y_i \in \mathbb{R}^m$ é seu rótulo correspondente, deseja-se encontrar, entre infinitas possibilidades, uma função f^* tal que $y^* = f^*(\mathbf{x})$, isto é, f^* deve ser capaz de produzir rótulos corretos para padrões desconhecidos independente e identicamente distribuídos (i.i.d.). A isso chama-se *capacidade de generalização*.

A função f^* deve ser aquela que melhor aproxima a resposta do Oráculo baseada em $\mathcal{D}$, ou seja, a função que melhor aproxima a dependência funcional desconhecida dos dados conhecidos. Para medir a qualidade de f^*, pode-se usar uma função de perda (ou custo) $l(y, f^*(\mathbf{x}))$. Uma função de custo muito comum é a *perda quadrática*:

$$l(y, f^*(\mathbf{x})) = (y - f^*(\mathbf{x}))^2. \tag{2.12}$$

O valor esperado para a perda é dado pelo risco esperado (Vapnik, 1999; Hastie et al., 2001; Castro, 2011):

$$R[f] = \int l(y, f^*(\mathbf{x}))p(\mathbf{x}, y)\,dy\,d\mathbf{x} = E[l(y, f^*(\mathbf{x}))], \tag{2.13}$$

no qual E é o valor esperado. Portanto, o objetivo do aprendizado é encontrar f^* que minimiza o risco $R[f]$ baseado no conjunto de treinamento $\mathcal{D}$.

Como a função de probabilidade conjunta $p(\mathbf{x}, y)$ usada para calcular o risco esperado é desconhecida, usa-se o risco empírico:

$$R_{emp}[f] = \frac{1}{N}\sum_{i=1}^{N} l(y_i, f^*(\mathbf{x}_i)). \tag{2.14}$$

O cálculo do risco empírico $R_{emp}[f]$ usa apenas os padrões de treinamento e converge para o risco esperado (Vapnik, 1999).

2.2.1.1 *Risco Empírico Regularizado*

É um fato bem conhecido no aprendizado supervisionado que a minimização do risco empírico por si só não garante um bom desempenho para dados desconhecidos (Scholkopf e Smola, 2001), porque o modelo pode estar sobreajustado (*overfitting*) aos dados de treinamento, especialmente se há ruído, resultando em uma função não suave. Nas RNAs, algumas abordagens têm sido desenvolvidas para melhorar a capacidade de generalização, tais como o controle da complexidade da rede com técnicas de poda e o controle da norma dos pesos usando otimização multiobjetivo.

Além de não haver garantia de bom desempenho de classificação, a minimização de $R_{emp}[f]$ pode levar a um problema mal condicionado (Scholkopf e Smola, 2001). O próprio problema de aprendizado, por ser um típico problema inverso, é mal condicionado (Velho, 2001). Nesse sentido, técnicas de regularização têm sido aplicadas com sucesso a problemas de aprendizado (Scholkopf e Smola, 2001). No contexto das RNAs, a regularização tem se mostrado útil para melhorar a capacidade de generalização das mesmas (Girosi et al., 1995).

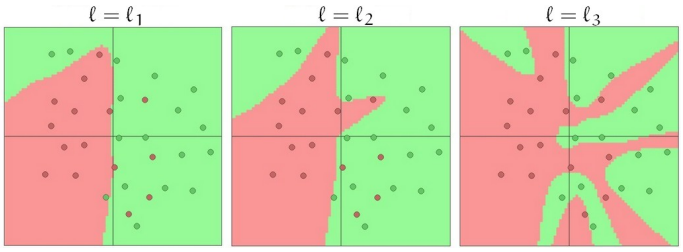
Assim, pode-se adicionar um termo de regularização ao risco empírico R_{emp} (2.14), de forma a se obter o Risco Empírico Regularizado R_{reg} (Scholkopf e Smola, 2001):

$$R_{reg}[f] = R_{emp}[f] + \lambda\Omega[f], \tag{2.15}$$

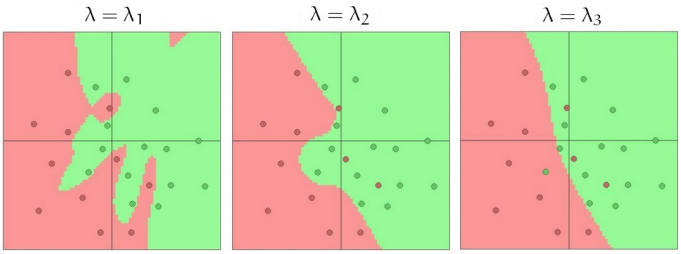
na qual λ é o parâmetro de regularização e $\Omega[f]$ normalmente é convexa. Por exemplo, uma escolha comum nas Máquinas de Vetores Suporte - *Support Vector Machines* (SVMs) é $\Omega[f] = \frac{1}{2}\|\mathbf{w}\|^2$ (Scholkopf e Smola, 2001).

As Figuras 2.2a e 2.2b apresentam, respectivamente, o efeito do sobreajuste e o efeito da regularização em uma RNA. É apresentado um problema de classificação binária, com os pontos de cada uma das duas classes sendo representados por uma cor diferente. São apresentadas também as superfícies de separação para cada caso. A Figura 2.2a apresenta o efeito do sobreajuste devido ao aumento no número de neurônios (ℓ): $\ell_3 > \ell_2 > \ell_1$ - à medida que o número de neurônios aumenta, a superfície de separação tende a perder a suavidade. Já na Figura 2.2b, obtida a partir da rede com ℓ_3 neurônios na camada escondida, pode-se observar o efeito da regularização, especialmente para o valor mais adequado do parâmetro de λ quando se deseja obter uma superfície com alto grau de suavidade: λ_3.

As Redes Neurais Regularizadas (*Regularized Neural Networks*) (Poggio e Girosi, 1990) são um exemplo clássico de regularização em RNAs. De acordo com Poggio e Girosi (1990), "a solução computada pela rede



(a) Superfícies de separação para valores crescentes de ℓ.



(b) Superfícies de separação para a rede com $\ell = \ell_3$, com diferentes valores para o parâmetro de regularização.

Figura 2.2: Efeitos do sobreajuste e da regularização em uma RNA (Karpathy, 2015).

regularizada é 'ótima' no sentido que minimiza um funcional que mede o quanto ela oscila, e isso elimina as soluções que interpolam perfeitamente os dados, mas oscilam mal onde não existem dados"[1]. Regularização em RNAs também pode ser encontrada nas Redes Neurais Regularizadas Generalizadas (*Generalized Regularized Neural Networks*) (Girosi et al., 1995), na base teórica das redes Função de Base Radial - *Radial Basis Function* (RBF), entre outras. Além disso, a teoria da regularização possui grande influência nas SVMs (Cortes e Vapnik, 1995).

2.3 MÁQUINAS DE APRENDIZADO EXTREMO

A forma tradicional de treinamento de redes Perceptron de Múltiplas Camadas - *Multi-Layer Perceptron* (MLP) - consiste em ajustar todos os pesos da rede, ou seja, ajustar as matrizes $\mathbf{Z}$ e $\mathbf{W}$ - Equações (2.3) e (2.4), respectivamente. Em geral, esse ajuste é baseado na minimização de um funcional que leva em consideração os sinais de erro obtidos na unidade de saída da rede, conforme Equação (2.8). Um caso especial das redes MLP são as SLFNs, as quais apresentam apenas uma camada escondida, conforme Figura 2.3.

O surgimento das máquinas de *kernel* (Cortes e Vapnik, 1995) proporcionou uma nova perspectiva para o problema de aprendizado, ao descrevê-lo em duas etapas: mapeamento não-linear dos padrões de

1 "the solution computed by the regularization network is 'optimal' in the sense that it minimizes a functional that measures how much it oscillates and this eliminates solutions that perfectly interpolate the data points but badly oscilate where there are no data."

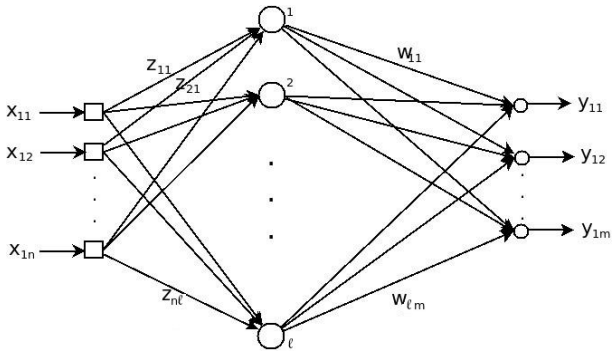


Figura 2.3: Uma Rede Neural de uma única Camada Alimentada Adiante - *Single-Layered Feedforward Neural Network* (SLFN).

entrada, por meio de um *kernel* previamente ajustado, seguido da estimação de uma separação linear no espaço de características de alta dimensão (Cristianini e Shawe-Taylor, 2000). Uma vez definidos os parâmetros do *kernel* e o mapeamento correspondente, um problema de otimização é então resolvido.

Esse mapeamento não-linear dos padrões de entrada em um espaço de alta dimensão tem seu fundamento no teorema de Cover (Cover, 1965), que, de modo simplificado, pode ser lido como (Haykin, 1994):

> "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."

Portanto, a projeção de um conjunto de dados não linearmente separáveis em um espaço de mais alta dimensão, por meio de uma transformação não-linear, aumenta a probabilidade de tornar as classes linearmente separáveis.

A Figura 2.4 apresenta um esquema geral para o treinamento em duas etapas, por meio do mapeamento do espaço de entrada para o espaço intermediário.

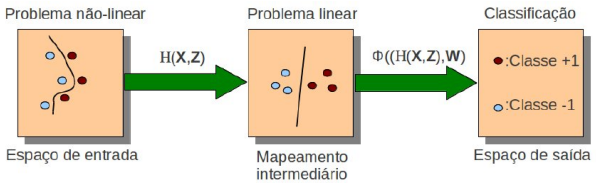


Figura 2.4: Esquema para treinamento em duas etapas, ilustrado para um problema de classificação binária (Braga, 2012).

Baseando-se nesse princípio, recentemente, uma nova forma de treinamento para SLFNs, chamada de Máquinas de Aprendizado Extremo - *Extreme Learning Machines* (ELMs) (Huang et al., 2004, 2006a), tem recebido atenção da comunidade científica. As ELMs utilizam o conceito de projeção aleatória (Miche et al., 2010a): os pesos da camada

escondida são escolhidos aleatoriamente, e não há treinamento para ajustá-los. O treinamento consiste em ajustar, de forma analítica, os pesos da camada de saída, conforme será visto na Seção 2.3.1. Como, devido ao Teorema de Cover, é necessário projetar os dados em um espaço de mais alta dimensão, a dimensão da camada escondida de uma ELM, isto é, o número de neurônios nessa camada, é consideravelmente maior do que o de uma SLFN treinada com um algoritmo como o de retropropagação de erros.

2.3.1 *Treinamento de ELMs*

Conforme mencionado anteriormente, os valores $\mathbf{z}_{ij}$ da matriz $\mathbf{Z}$ - pesos da camada escondida - de uma ELM são atribuídos de forma aleatória, segundo uma distribuição uniforme. Após o cálculo da matriz $\mathbf{U}$ (Equação 2.5), segue o cálculo da matriz $\mathbf{H}$ (Equação 2.6). Esse cálculo é, simplesmente, a aplicação da função de ativação h sobre a matriz $\mathbf{U}$. É importante destacar que, ao contrário de alguns algoritmos de treinamento de RNAs, como o EBP, não é necessário que a função de ativação de uma ELM seja diferenciável, além de ser possível utilizar funções de ativação diferentes em uma mesma rede (Miche et al., 2010a). Finalizada a projeção no espaço intermediário, a matriz $\mathbf{W}$ de pesos da camada escondida é calculada pela pseudoinversa de Moore-Penrose (Serre, 2002).

Portanto, o treinamento de uma ELM pode ser resumido como segue. Considere $\mathbf{X}$ como sendo os dados de treinamento e $\mathbf{Y}$ seus respectivos rótulos:

- *Primeira etapa* - mapeamento intermediário, com objetivo de linearizar o problema:

$$\mathbf{H} = \Psi(\mathbf{X}, \mathbf{Z}) \tag{2.16}$$

- *Segunda etapa* - solução do problema linearizado:

$$\hat{\mathbf{Y}} = \phi(\mathbf{H}, \mathbf{W}) \tag{2.17}$$

A primeira etapa consiste, então, em encontrar a saída $\mathbf{H}$ da camada escondida por meio da projeção, que inclui a aplicação da função de ativação:

$$\mathbf{H} = h(\mathbf{XZ}) \tag{2.18}$$

A segunda etapa envolve encontrar a matriz de pesos de saída $\mathbf{W}$. Tem-se que:

$$\mathbf{Y} = \mathbf{HW} \tag{2.19}$$

Se $N = p$, ou seja, se o número de padrões fosse sempre igual ao número de neurônios da camada escondida (e $\mathbf{H}$ fosse não-singular), poderia ser feito:

$$\mathbf{W} = \mathbf{H}^{-1}\mathbf{Y} \tag{2.20}$$

Nesse caso, a rede aprenderia exatamente todos os N padrões de treinamento (Huang et al., 2006a). Como muito raramente tem-se $N = p$ - comumente, $N \gg p$ -, o uso da inversa $\mathbf{H}^{-1}$ não é possível, assim como não é interessante que a rede aprenda exatamente os padrões de treinamento, já que isso pode provocar perda da capacidade de generalização. Por isso, utiliza-se a inversa generalizada de Moore-Penrose (pseudoinversa), $\mathbf{H}^{\dagger}$, fazendo, então:

$$\mathbf{W} = \mathbf{H}^{\dagger}\mathbf{Y}. \tag{2.21}$$

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

$$\hat{\mathbf{Y}} = \mathbf{H}\mathbf{W}. \tag{2.22}$$

O erro de treinamento pode ser calculado por:

$$\mathbf{E} = \mathbf{Y} - \hat{\mathbf{Y}}. \tag{2.23}$$

Após o treinamento, podem ser apresentados padrões que a rede desconhece - padrões de teste. Seja $\mathbf{X}_{test}$ o conjunto dos padrões de teste. Para obter a saída $\hat{\mathbf{Y}}_{test}$, é necessário obter a matriz $\mathbf{H}_{test}$, ou seja, obter o mapeamento dos dados de teste no espaço da camada escondida:

$$\mathbf{H}_{test} = h(\mathbf{X}_{test}\mathbf{Z}). \tag{2.24}$$

Por fim, faz-se:

$$\hat{\mathbf{Y}}_{test} = \mathbf{H}_{test}\mathbf{W}, \tag{2.25}$$

que é a saída da rede para o conjunto padrões de teste $\mathbf{X}_{test}$.

2.3.1.1 *Características das ELMs*

As ELMs apresentam algumas características importantes. A primeira delas é a *rapidez do treinamento*: por não envolver um processo iterativo, como no caso do EBP e de outros algoritmos de treinamento de MLPs, o tempo necessário para o treinamento costuma ser significativamente menor que aquele despendido por outras técnicas. Por outro

lado, como, em geral, o número de neurônios na camada escondida das ELMs é maior, essa rapidez de treinamento não se reflete diretamente no tempo de teste, isto é, pela alta dimensionalidade da rede, o tempo de resposta para novos padrões em uma ELM costuma ser maior do que, por exemplo, em uma rede treinada por EBP. Outra característica interessante é a *simplicidade de configuração*, já que o único parâmetro relevante é o número de neurônios da camada escondida. A boa *capacidade de generalização* também é uma característica importante das ELMs.

2.3.1.2 *Variações das ELMs*

Desde o seu surgimento, diversas versões de ELMs têm sido desenvolvidas, mostrando a mesma ser um tema bastante ativo na comunidade científica. A seguir, são apresentadas algumas dessas versões.

Em Huang et al. (2005), é apresentada uma versão online da ELM - ELM Sequencial Online - *Online Sequential Extreme Learning Machine* (OS-ELM), na qual o algoritmo original é adaptado para permitir a inclusão de novos padrões de treinamento de forma *online* e sequencial, sem que seja necessário repetir os cálculos feitos anteriormente para os pesos da camada de saída. Já a *Ensemble of Online Sequential Extreme Learning Machines* (EOS-ELM) (Lan et al., 2009) apresenta um *ensemble* de várias OS-ELM com o mesmo número de nós e com a mesma função de ativação na camada escondida, com objetivo de reduzir a falta de estabilidade apresentada por uma rede isolada, já que o resultado do *ensemble* é a média dos valores de cada OS-ELM que o compõe.

A ELM Incremental - *Incremental Extreme Learning Machine* (I-ELM) (Huang et al., 2006b) possibilita a construção incremental de ELMs: inicialmente, são informados o número máximo de neurônios e o erro tolerado e, a cada passo, um novo neurônio é adicionado à camada escondida, tem seu peso z atribuído de forma aleatória e seu peso w calculado. O processo é repetido até que seja obtido um erro menor ou igual ao erro informado, ou o número máximo de neurônios seja alcançado.

A ELM Evolucionária - *Evolutionary ELM* (E-ELM) (Zhu et al., 2005) utiliza Evolução Diferencial - *Differential Evolution* (DE) para encontrar o conjunto ótimo dos pesos e termos de polarização da camada escondida, possibilitando obter uma rede mais compacta e, assim, melhorar o tempo de teste da rede, já que a mesma fica com menos neurônios pouco relevantes - a presença desse tipo de neurônios é comum em ELMs. Essa seleção de modelo é feita a partir de uma função de aptidão (*fitness function)*, a qual é calculada por Raiz do Erro Quadrático Médio - *Root Mean Squared Error* (RMSE) sobre um conjunto de validação. Quando a diferença entre aptidões de dois indivíduos é muito pequena, é retido o indivíduo que possui a menor norma dos pesos de saída.

Em (Feng et al., 2009) é proposta uma abordagem para determinar, de forma automática, o número de nós na camada escondida de uma ELM - são adicionados nós aleatórios à rede, isoladamente ou em grupos, atualizando os pesos de saída de forma incremental. Os novos nós são adicionados até que se alcance a acurácia desejada.

Em (Yin et al., 2009), é apresentada uma abordagem para selecionar, de um *pool*, nós para a camada escondida. Essa abordagem utiliza, ainda, um critério que representa um compromisso entre o desempenho e o número de parâmetros da rede para determinar automaticamente o número de nós na camada escondida.

No contexto de SVM e Regressão por Vetores Suporte - *Support Vector Regression* (SVR), (Frénay e Verleysen, 2010) e (Frénay e Verleysen, 2011) apresentam a construção de *kernels* baseados em ELM, os quais possuem a vantagem de não necessitarem de ajustes de parâmetros caso seja selecionado um número ℓ de neurônios suficientemente grande para a camada escondida da ELM (os autores afirmam que $\ell > 1000$ é suficiente). Isso reduz significativamente o custo computacional envolvido na construção de *kernels*. Em (Huang et al., 2012), a ELM também é estendida para lidar com aprendizado de *kernel*, apresentando as formas pelas quais os conhecidos algoritmos SVM de Mínimos Quadrados - *Least Squares Support Vector Machine* (LS-SVM) (Suykens e Vandewalle, 1999) e *Proximal Support Vector Machine* (PSVM) (Mangasarian e Wild, 2001) podem ser simplificados, por meio da remoção do termo de polarização b (*bias*), e os algoritmos resultantes unificados como ELMs. Além disso, ao contrário dos algoritmos LS-SVM e PSVM originais, as ELMs conseguem tratar naturalmente problemas de classificação multiclasses e problemas de regressão.

Uma abordagem multiobjetivo para o treinamento de ELMs, visando obter redes compactas com boa capacidade de generalização, é apresentada em (Mao et al., 2012). Os neurônios são adicionados um a um à camada escondida e, a cada passo, o algoritmo de otimização multiobjetivo é usado para selecionar os pesos de entrada ótimos pela minimização do limite *Leave-One-Out* (LOO) e da norma dos pesos de saída.

2.3.2 *Regularização em ELMs*

A ELM original não impõe qualquer controle sobre a suavidade da resposta da rede. Conforme visto, técnicas de regularização podem ser usadas para controlar a suavidade e, consequentemente, incrementar o desempenho de classificação da ELM. Como a ELM pode ser facilmente modificada para incluir um termo de regularização, modificações para fazer isso durante a estimação dos pesos da camada de saída têm sido propostas.

Em (Deng et al., 2009), é proposta uma ELM regularizada que é, essencialmente, uma ELM com penalidade L_2, ou seja, regularização de

Tikhonov, com a possibilidade de ponderar a soma dos quadrados para lidar com interferência de *outliers*. A formulação desenvolvida inclui um parâmetro escalar γ que permite ajustar a proporção do risco empírico e do risco estrutural. Além disso, as variáveis de erro ξ_i são ponderadas por pesos v_i, para enfraquecer a interferência de *outliers*, de forma que a função de custo $\|\xi\|^2$ é estendida como $\|D\xi\|^2$. O cálculo dos pesos da camada de saída $\mathbf{W}$ é feito com a fórmula:

$$\mathbf{W} = \left(\frac{I}{\gamma} + \mathbf{H}^T D^2 \mathbf{H}\right)^{\dagger} \mathbf{H}^T D^2 \mathbf{Y}, \tag{2.26}$$

na qual $\mathbf{Y}$ é a matriz de rótulos (alvos), D é uma matriz com os pesos $v_1, v_2, \ldots, v_N$, com $\ell \ll N$. A versão com $D = \mathbf{I}$ (matriz identidade) é chamada ELM não ponderada regularizada (*unweighted regularized* ELM):

$$\mathbf{W} = \left(\frac{I}{\gamma} + \mathbf{H}^T \mathbf{H}\right)^{\dagger} \mathbf{H}^T \mathbf{Y}. \tag{2.27}$$

O parâmetro γ é ajustado usando validação cruzada.

A ELM de Taxa de Erro Total - *Total Error Rate ELM* (TER-ELM) (Toh, 2008) é uma abordagem que inclui o número total de padrões positivos e negativos (m^+ e m^-, respectivamente) em sua formulação. Assim, é definida uma matriz de ponderação diagonal específica para cada uma das duas classes, $\mathbf{M}^+ = \mathrm{diag}(0, \ldots, 0, 1/m^+, \ldots, 1/m^+)$ e $\mathbf{M}^- = \mathrm{diag}(1/m^-, \ldots, 1/m^-, 0, \ldots, 0)$; $y^+ = (\tau + \eta)\mathbf{1}_+$ e $y^- = (\tau - \eta)\mathbf{1}_-$, na qual τ e η são limiar e viés comuns para todos os padrões. A solução é:

$$\mathbf{W} = \left(\mathbf{H}^T \mathbf{M} \mathbf{H}\right)^{-1} \mathbf{H}^T \mathbf{M} \mathbf{Y}, \tag{2.28}$$

na qual $\mathbf{M} = \mathbf{M}^+ + \mathbf{M}^-$ e os elementos de $\mathbf{H}$ e $\mathbf{Y}$ são ordenados de acordo com as amostras positivas e negativas das duas classes.

Em redes com $\ell \gg N$, para melhorar a estabilidade numérica, uma regularização por decaimento dos pesos é incluída, e a fórmula torna-se:

$$\mathbf{W} = (\mathbf{H}^T \mathbf{M} \mathbf{H} + \lambda \mathbf{I})^{-1} \mathbf{H}^T \mathbf{M} \mathbf{Y}. \tag{2.29}$$

Em todos os casos, há parâmetros a serem ajustados: λ e as matrizes $\mathbf{M}^+$ e $\mathbf{M}^-$. Os autores afirmam que incluíram m^+ e m^- como parte dos hiperparâmetros do modelo de forma que eles possam ser ajustados para lidar com diferenças nos tamanhos das distribuições das classes positiva e negativa, assim como com erros decorrentes da aproximação quadrática.

Em (Martínez-Martínez et al., 2011), os autores apresentam um método de regularização para a ELM no qual a ideia principal é identificar

o grau de relevância do peso que conecta o k-ésimo nó da camada escondida com a saída usando métodos de regressão: *lasso*, *ridge regression* e *elastic net*. O problema de minimização é:

$$\min_{(\beta_0,\boldsymbol{\beta})\in\mathbb{R}^{\ell}} \left[\frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \mathbf{h}_i^T \boldsymbol{\beta})^2 + \lambda P_\alpha(\boldsymbol{\beta}), \right], \tag{2.30}$$

no qual

$$P_\alpha(\boldsymbol{\beta}) = \sum_{j=1}^{\ell} \left[\frac{1}{2}(1-\alpha)\boldsymbol{\beta}_j^2 + \alpha|\boldsymbol{\beta}_j| \right], \tag{2.31}$$

em que $[\beta_0 \boldsymbol{\beta}]$ é uma coluna de $\mathbf{Z}$, e o parâmetro α é um compromisso (*trade-off*) entre a penalidade da *ridge regression* ($\alpha = 0$) e a penalidade *lasso* ($\alpha = 1$).

O parâmetro de regularização λ é selecionado usando o Critério de Informação Bayesiana - *Bayesian Information Criterion* (BIC) (Schwarz, 1978). Para cada valor de λ, existem K modelos - cada um com um número distinto de neurônios na camada escondida. O BIC seleciona modelos com um compromisso entre acurácia e tamanho da rede, introduzindo um termo de penalidade para o número de parâmetros do modelo. Após todos os modelos serem gerados, o BIC é calculado para cada modelo correspondente a cada valor de λ. O λ selecionado é aquele com o valor BIC mínimo.

Em Miche et al. (2011), os autores apresentam a ELM Podada de forma Ótima com Regularização de Tikhonov - *Tikhonov Regularized Optimally Pruned ELM* (TROP-ELM), a qual é um incremento à ELM Podada de forma Ótima - *Optimally Pruned Extreme Learning Machine* (OP-ELM) (Miche et al., 2008, 2010a). Nesta, uma penalidade L_1 (*lasso*) é usada para ranquear os neurônios da camada escondida, e o problema de minimização é:

$$\min_{(\lambda,\hat{\mathbf{z}})} \left[\sum_{i=1}^{N} (y_i - \mathbf{x}_i \hat{\mathbf{z}})^2 + \lambda \sum_{j=1}^{n} |\hat{z}_j| \right]. \tag{2.32}$$

Na TROP-ELM, após aplicar a penalidade L_1 para selecionar neurônios, é aplicada uma penalidade L_2 (regularização de Tikhonov) nos pesos da camada de saída. Tanto na OP-ELM quanto na TROP-ELM há parâmetros de regularização que devem ser ajustados: λ para OP-ELM (2.32) e λ_2 introduzido pela penalidade L_2 na TROP-ELM. Uma abordagem similar, que combina estimativa de distância par-a-par para lidar com dados ausentes é apresentada em (Yu et al., 2013).

A versão apresentada por Huang et al. (2012) também é uma versão regularizada da ELM. Esta versão adiciona um termo de regularização,

C, ao cálculo da pseudoinversa. Caso o número de amostras de treinamento seja menor que o número de neurônios da camada escondida ($N < \ell$), os pesos da camada de saída β são calculados como segue:

$$\mathbf{W} = \mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{Y}. \tag{2.33}$$

Se $N \gg \ell$, o cálculo de β é:

$$\mathbf{W} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^T\mathbf{H} \right)^{-1} \mathbf{H}^T\mathbf{Y}, \tag{2.34}$$

o qual é o mesmo de (2.27). As duas equações (2.33) e (2.34) levam ao mesmo resultado, mas quando o conjunto de dados de treinamento é grande, a Equação (2.34) pode ser usada para reduzir custos computacionais. Como nas demais versões de ELMs regularizadas, esta versão também possui um parâmetro que necessita de ser ajustado (C), o que é feito, normalmente, com validação cruzada.

2.4 APRENDIZADO SEMISSUPERVISIONADO

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

A Figura 2.5 apresenta o aprendizado semissupervisionado visto da perspectiva do aprendizado supervisionado. O Gerador de Dados - *Data Generator* (DG) fornece os dados X de acordo com a distribuição $P(x)$. O agente de rotulação, também conhecido como Oráculo (O), o qual conhece uma aproximação da distribuição condicional $P(y|x)$, fornece rótulos para os N_L padrões $x \in D_L$. Também estão disponíveis um conjunto de parâmetros Θ e N_U padrões amostrados, dos quais é extraído o modelo não-supervisionado (M_U), gerando assim um estimador não-supervisionado (E_U). Diferentemente do aprendizado supervisionado, no qual o modelo M é gerado a partir dos padrões rotulados apenas, no aprendizado semissupervisionado, a geração do modelo M_{SS}, além de utilizar o conjunto A de parâmetros α, pode levar em consideração a densidade de probabilidade estimada pelo estimador E_U. O Estimador Semissupervisionado (E_{SS}), então, tem a capacidade de prever o valor de y^* a partir de um novo padrão x^*, ou seja, no caso da classificação, prever a qual classe pertence esse padrão.

Já do ponto de vista do aprendizado não-supervisionado, como visto na Figura 2.6, informações sobre dados rotulados, provenientes do Oráculo, são adicionadas ao modelo M, ou seja, a geração do modelo conta com essa informação adicional, e a busca por agrupamentos deve ser consistente com os dados rotulados.

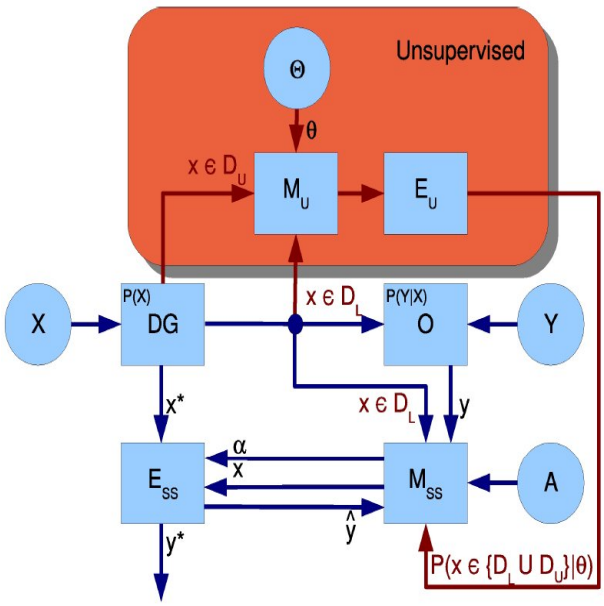


Figura 2.5: Esquema Geral do Aprendizado Semissupervisionado do Ponto de Vista do Aprendizado Supervisionado (Braga, 2012).

2.4.1 *Premissas do Aprendizado Semissupervisionado*

Dado que o conjunto de dados rotulados é escasso, é necessário que o aprendizado semissupervisionado seja baseado em algumas premissas ou hipóteses (*assumptions*). Por exemplo, considere o problema de classificação binária apresentado na Figura 1.1. Caso não seja considerada premissa alguma, não é possível afirmar que o resultado obtido pela classificação semissupervisionada (Figura 1.1b) é melhor que aquele obtido pela classificação supervisionada (Figura 1.1a). Por outro lado, é possível observar, nesta figura, uma região de baixa densidade entre dois grupos de dados. Assumindo, *a priori*, que a superfície de separação está em uma região de baixa densidade, é possível, então, afirmar a vantagem do método semissupervisionado. As principais premissas são as que seguem (Chapelle et al., 2006) (os algoritmos baseiam-se em uma ou mais dessas premissas):

- **Premissa de Suavidade - *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.

- **Premissa de Agrupamento - *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.

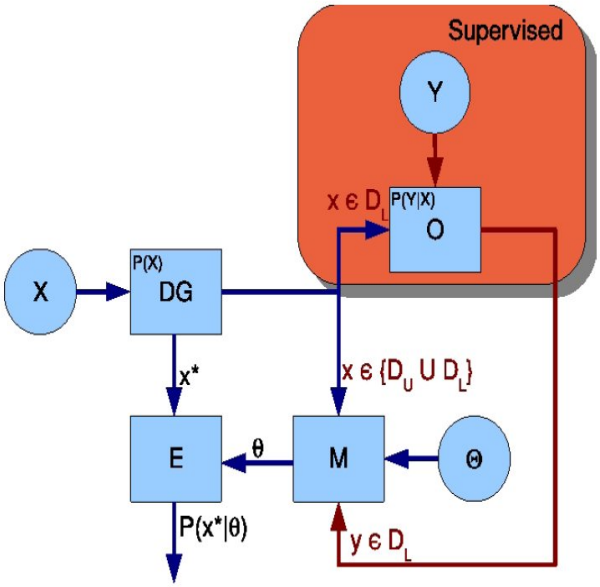


Figura 2.6: Esquema Geral do Aprendizado Semissupervisionado do Ponto de Vista do Aprendizado Não-Supervisionado (Braga, 2012).

- ***Manifold Assumption***: Os dados originais estão imersos em um espaço de dimensão menor, ou seja, é possível reduzir a dimensão sem perda de informação.

2.4.2 *Máquina de Aprendizado Extremo Semissupervisionada - Semi-Supervised ELM*

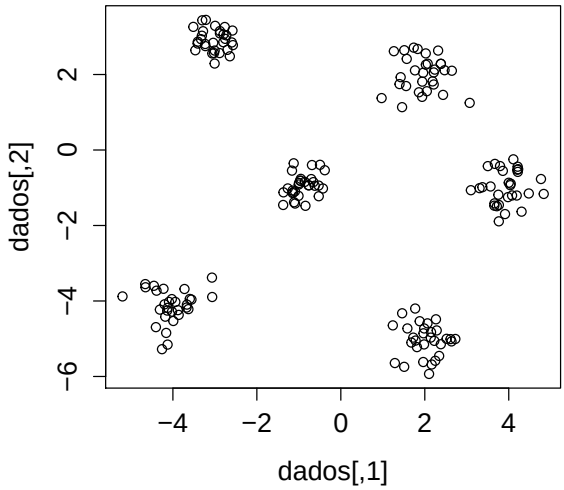
Recentemente, foram desenvolvidas versões semissupervisionadas da ELM. A versão apresentada em (Liu et al., 2011), chamada de Máquina de Aprendizado Extremo Semissupervisionada - *Semi-Supervised ELM* (SELM), é aplicada em localização baseada em redes wi-fi, e tem sua abordagem baseada na construção 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 (Belkin et al., 2004), 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 (Li et al., 2013), também chamada de SELM, é baseada em cotreinamento (*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 *label noise* (ruído de rótulo, ou seja, amostras rotuladas incorretamente), mas versões robustas da rede ELM (Barros e Barreto, 2013),

por lidarem bem com esse tipo de ruído, são boas candidatas para *co-training*.

2.5 MATRIZES DE AFINIDADE

Uma matriz de afinidade (ou de similaridade ou, ainda, de proximidade) $\mathbf{P}_{N\times N}$, na qual N é o número de padrões, é tal que cada elemento p_{ij} possui uma medida de afinidade entre os padrões $\mathbf{x}_i$ e $\mathbf{x}_j$. A Figura 2.8 apresenta a matriz de similaridade de cossenos obtida para os dados apresentados na Figura 2.7. A diagonal principal da matriz apresenta os valores de afinidade intra-grupos, os quais são elevados quando comparados com os valores de fora da diagonal, que representam valores de afinidade inter-grupos.



so

Figura 2.7: Dados sintéticos, 6 grupos - distribuição normal bivariada.

A afinidade entre dois padrões $\mathbf{x}$ e $\mathbf{y} \in \mathbf{X}$ é uma medida que quantifica a dependência entre ambos (Goshtasby, 2012). Medidas de afinidade têm sido utilizadas em diversas aplicações de reconhecimento de padrões, incluindo classificação, regressão, agrupamento, seleção de características, entre outras (Bandyopadhyay e Saha, 2013). Dado que a estrutura do conjunto de dados pode ser explorada usando medidas de similaridade (Bandyopadhyay e Saha, 2013), pode-se afirmar que essas medidas permitem obter informações importantes sobre a estrutura dos dados.

Uma medida de similaridade em um conjunto $\mathbf{X}$ é definida como (Theodoridis e Koutroumbas, 2008):

$$s : \mathbf{X} \times \mathbf{X} \rightarrow \mathcal{R}$$

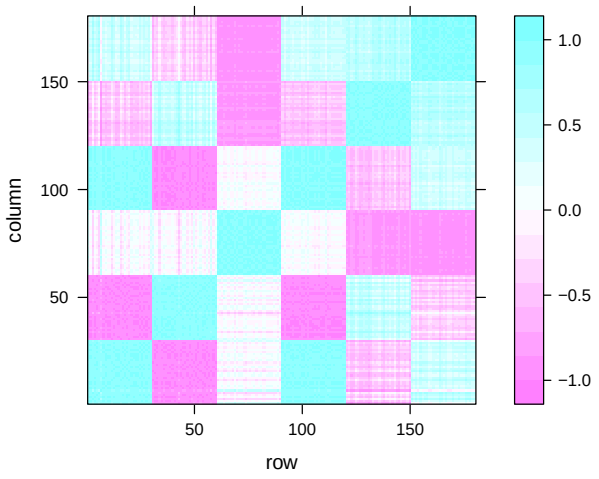


Figura 2.8: Matriz de similaridade de cossenos obtida a partir dos dados apresentados na Figura 2.7.

tal que

$$\exists s_0 \in \mathcal{R} : -\infty < s(\mathbf{x}, \mathbf{y}) \leqslant s_0 < +\infty, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbf{X} \tag{2.35}$$

$$s(\mathbf{x}, \mathbf{x}) = s_0, \quad \forall \mathbf{x} \in \mathbf{X} \tag{2.36}$$

e

$$s(\mathbf{x}, \mathbf{y}) = s(\mathbf{y}, \mathbf{x}), \quad \forall \mathbf{x}, \mathbf{y} \in \mathbf{X}. \tag{2.37}$$

Uma medida de similaridade S é considerada uma métrica se produz um valor maior à medida que a correspondência entre os padrões aumenta. Por outro lado, uma medida de dissimilaridade D é considerada uma métrica caso produza um valor maior à medida que a correspondência entre os padrões diminui (Goshtasby, 2012). Medidas de distância são consideradas medidas de dissimilaridade (a distância entre determinado padrão e ele mesmo é zero). Caso as medidas de similaridade s_{ij} e dissimilaridade d_{ij} entre dois padrões i e j sejam normalizadas, elas podem ser relacionadas como segue (Bandyopadhyay e Saha, 2013):

$$s_{ij} = 1 - d_{ij}$$

Neste trabalho, tanto medidas de similaridade como de dissimilaridade são tratadas, genericamente, como medidas de afinidade.

Alguns exemplos de medidas de afinidade são: *Coeficiente de Correlação de Pearson*, *Medida de Tanimoto*, *Distâncias Euclidiana*, *City Block*, *de Chebyshev* e *de Mahalanobis* (Bandyopadhyay e Saha, 2013; Cha, 2007). Dada a grande quantidade de medidas de afinidade disponíveis, identificar a mais adequada de acordo com a aplicação é um problema importante.

O produto escalar (produto interno no espaço euclidiano) também é uma medida de afinidade, em especial o produto interno *Kernel*, muito utilizado em Aprendizado de Máquina na construção de Máquinas de Vetor de Suporte (*Support Vector Machines* - SVMs) (Cortes e Vapnik, 1995).

2.5.1 *Kernels*

Um *kernel* ou *produto interno kernel* pode ser definido por (Haykin, 1994):

$$K(\mathbf{x}, \mathbf{x}_i) = \varphi^T(\mathbf{x})\varphi(\mathbf{x}_i). \tag{2.38}$$

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 φ" (de Andrade Queiroz, 2009).

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

$$k(\mathbf{x}, \mathbf{x}_i) = e^{\frac{-\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}}, \tag{2.39}$$

onde σ^2 é a largura (raio) comum a todos os *kernels*, especificada *a priori* pelo usuário (Haykin, 1994). Um *kernel* gerado a partir da Equação (2.39) 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:

$$k(\mathbf{x}, \mathbf{x}_i) = \mathbf{x}^T\mathbf{x}_i. \tag{2.40}$$

2.5.2 *Similaridade de Cossenos*

Outra medida de afinidade importante, a qual será utilizada neste trabalho, é a Similaridade de Cossenos, definida como o produto interno normalizado (Cha, 2007). Seu nome decorre do fato de ela medir o ân-

gulo entre dois vetores. Cada elemento p_{ij} de uma matriz $N \times N$ de Similaridade de Cossenos P é assim calculado:

$$p_{ij} = \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\| \left\|\mathbf{x}_j\right\|}, \tag{2.41}$$

em que $\|\cdot\|$ é a norma L_2 ou norma euclidiana.

Como a medida é normalizada, temos que $-1 \leqslant p_{ij} \leqslant 1$, com $p_{ii} = 1$, ou seja, a diagonal principal da matriz de similaridade de cossenos é igual a 1.

2.6 CONCLUSÃO DO CAPÍTULO

Neste capítulo foi apresentada uma revisão bibliográfica sobre os principais assuntos deste texto: Regularização, Máquinas de Aprendizado Extremo (*Extreme Learning Machines*), Aprendizado Semissupervisionado e Matrizes de Afinidade. Com relação à Regularização, foi apresentada a visão geral e sua relação com o aprendizado de máquina. Para as ELMs, foi apresentado seu desenvolvimento teórico, algumas variações presentes na literatura, incluindo versões que aplicam regularização. Com relação ao Aprendizado Semissupervisionado e às Matrizes de Afinidade, foi apresentada uma visão geral, com conceitos básicos.

3

REGULARIZAÇÃO DE EXTREME LEARNING MACHINES COM MATRIZES DE AFINIDADE

Este capítulo apresenta o desenvolvimento formal do método de regularização para ELMs utilizando matrizes de afinidade, o qual é denominado ELM Regularizada com Matrizes de Afinidade - *Affinity Matrix Regularized ELM* (AMR-ELM). A regularização é feita por meio da interferência no treinamento da ELM, introduzindo uma matriz de afinidade **P**. Além do efeito de regularização de Tikhonov obtido, cuja importância já foi ressaltada nos capítulos anteriores, o método também possibilita que a ELM seja utilizada para classificação semissupervisionada, tendo em vista que, como consequência do mesmo, é feita uma estimativa dos rótulos ausentes. São apresentados ainda os algoritmos de projeção de padrões de treinamento e de teste, além da análise de complexidade computacional do método, em comparação com a ELM original.

3.1 INTRODUÇÃO

Conforme visto na Seção 2.3, o treinamento de uma ELM é feito em duas etapas. A partir disso, é possível visualizar uma forma de inserir uma alteração no valor alimentado adiante, por meio de uma via externa. Essa alteração insere, na saída da camada escondida, informações sobre a estrutura dos dados, contidas em uma matriz **P**, a qual pode ser uma matriz de afinidade. A Figura 3.1 apresenta essa ideia: os padrões de entrada **X** alimentam a rede e, paralelamente, são calculadas a saída **H** da camada escondida, conforme Equação 2.6, e a matriz de afinidade, por meio de uma função de Kernel K. Posteriormente, as matrizes **P** e **H** são multiplicadas, gerando a matriz $\mathbf{H}'$, a qual é a saída da camada escondida modificada pela afinidade. Por fim, a matriz $\mathbf{H}'$ é utilizada para o cálculo da matriz de pesos da camada de saída, $\mathbf{W}'$, permitindo assim obter a estimativa dos rótulos, $\hat{\mathbf{Y}}$.

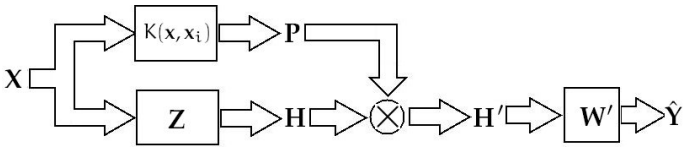


Figura 3.1: Processo de inserção de informação estrutura dos dados em uma ELM.

No método a ser proposto, o qual chamaremos de AMR-ELM, a regularização é realizada modificando-se o algoritmo da ELM original, inserindo a informação *a priori* expressa por uma matriz de afinidade - por exemplo, uma matriz de similaridade de cossenos (2.41).

3.2 DESENVOLVIMENTO DO MÉTODO

O Algoritmo 1 apresenta o procedimento proposto, o qual transforma a matriz de saída da camada escondida **H** usando uma matriz de afinidade **P**.

Algoritmo 1: Projeção de padrões de treinamento.

Entrada: **XTrain**: padrões de treinamento
YTrain: rótulos dos padrões de treinamento, incluindo termo de polarização
Z: matriz de pesos da camada escondida, gerada aleatoriamente
Saída: $\mathbf{H}'$: matriz com a projeção dos padrões na camada escondida, transformada pela matriz de afinidade

$\mathbf{P} \leftarrow \texttt{afinidade}(\mathbf{XTrain})$;
$\mathbf{H} \leftarrow \texttt{funcaoAtivacao}(\mathbf{XTrain} * \mathbf{Z})$;
$\mathbf{H}' \leftarrow \mathbf{P} * \mathbf{H}$;

A matriz de saída da camada escondida, **H**, é transformada para que possa incluir informação estrutural dos dados de treinamento, informação esta representada por **P**. A matriz resultante, $\mathbf{H}'$, é definida a seguir.

Definição 3.1. *A matriz* $\mathbf{H}'$*, a qual é a de saída da camada escondida de uma ELM modificada por uma matriz de afinidade* **P***, é definida como:*

$$\mathbf{H}' = \mathbf{PH}. \tag{3.1}$$

A partir da matriz $\mathbf{H}'$, pode ser calculada a matriz de pesos da camada de saída, $\mathbf{W}'$, conforme proposição a seguir.

Proposição 3.2. *Sejam a matriz de afinidade* **P** *e a matriz* $\mathbf{H}'$*, definida na Equação 3.1. A matriz de pesos da camada de saída* $\mathbf{W}'$ *pode ser calculada como:*

$$\mathbf{W}' = (\mathbf{H}'^{\mathsf{T}}\mathbf{H}')^{-1}\mathbf{H}'^{\mathsf{T}}\mathbf{Y}, \tag{3.2}$$

Essa fórmula equivale ao cálculo da matriz de pesos da camada de saída da ELM original, apenas utilizando $\mathbf{H}'$ *no lugar de* **H**.

É possível reescrever a matriz $\mathbf{H}'$ em termos de uma multiplicação de matrizes, na qual uma delas é diagonal, seguida de uma adição, conforme proposição a seguir.

Proposição 3.3. $\mathbf{H}'$ *pode ser decomposta em:*

$$\mathbf{H}' = \mathbf{AH} + \mathbf{B}, \tag{3.3}$$

em que

$$\mathbf{A} = \begin{bmatrix} p_{11} & 0 & \dots & 0 \\ 0 & p_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & p_{NN} \end{bmatrix}_{N \times N} \tag{3.4}$$

e

$$\mathbf{B} = \begin{bmatrix} \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq 1}} p_{1k} h_{k1} & \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq 1}} p_{1k} h_{k2} & \dots & \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq 1}} p_{1k} h_{k\ell} \\ \vdots & \vdots & \ddots & \vdots \\ \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq N}} p_{Nk} h_{k1} & \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq N}} p_{Nk} h_{k2} & \dots & \sum\limits_{\substack{1 \leqslant k \leqslant N, \\ k \neq N}} p_{Nk} h_{k\ell} \end{bmatrix}_{N \times \ell}, \tag{3.5}$$

pois cada elemento h'_{ij} *da matriz* $\mathbf{H}'$ *é definido como:*

$$h'_{ij} = \sum_{k=1}^{N} p_{ik} h_{kj}, \tag{3.6}$$

que pode ser reescrita como:

$$h'_{ij} = p_{ii} h_{ij} + \sum_{k=1, k \neq i}^{N} p_{ik} h_{kj}. \tag{3.7}$$

No caso de a matriz diagonal envolvida na multiplicação ser a matriz identidade, é possível simplificar o cálculo, resultando apenas em uma adição de matrizes, conforme proposição a seguir.

Proposição 3.4. *Caso a matriz* $\mathbf{P}$ *seja simétrica e normalizada, ou seja,* $p_{ij} = 1$ *se* $i = j$, $p_{ij} = p_{ji}$ *e* $-1 \leqslant p_{ij} \leqslant 1$ *para* $i \neq j$, *temos que a Equação 3.3 pode ser escrita como:*

$$\mathbf{H}' = \mathbf{H} + \mathbf{B}, \tag{3.8}$$

pois $\mathbf{A} = \mathbf{I}_{N \times N}$.

O desenvolvimento da Equação 3.2 possibilita observar uma equivalência com a regularização de Tikhonov (Tikhonov, 1963). Isso é enunciado no teorema a seguir.

Teorema 3.5. *O cálculo de* $\mathbf{W}'$ *equivale a uma regularização de Tikhonov.*

Demonstração.

$$\begin{aligned}\mathbf{W}' &= ((\mathbf{H}+\mathbf{B})^{\mathsf{T}}(\mathbf{H}+\mathbf{B}))^{-1}\mathbf{H}'^{\mathsf{T}}\mathbf{Y} \\ &= (\mathbf{H}^{\mathsf{T}}\mathbf{H}+\mathbf{H}^{\mathsf{T}}\mathbf{B}+\mathbf{B}^{\mathsf{T}}\mathbf{H}+\mathbf{B}^{\mathsf{T}}\mathbf{B})^{-1}(\mathbf{P}\mathbf{H})^{\mathsf{T}}\mathbf{Y} \\ &= (\mathbf{H}^{\mathsf{T}}\mathbf{H}+\Lambda)^{-1}\mathbf{H}^{\mathsf{T}}\mathbf{P}^{\mathsf{T}}\mathbf{Y} \\ &= (\mathbf{H}^{\mathsf{T}}\mathbf{H}+\Lambda)^{-1}\mathbf{H}^{\mathsf{T}}\mathbf{Y}', \end{aligned} \tag{3.9}$$

na qual $\mathbf{Y}' = \mathbf{P}^{\mathsf{T}}\mathbf{Y} = \mathbf{P}\mathbf{Y}$ e $\Lambda = \mathbf{H}^{\mathsf{T}}\mathbf{B}+\mathbf{B}^{\mathsf{T}}\mathbf{H}+\mathbf{B}^{\mathsf{T}}\mathbf{B}$. Como $\mathbf{B} = \mathbf{H}'-\mathbf{H}$:

$$\begin{aligned}\Lambda &= \mathbf{H}^{\mathsf{T}}(\mathbf{H}'-\mathbf{H})+(\mathbf{H}'-\mathbf{H})^{\mathsf{T}}\mathbf{H}+(\mathbf{H}'-\mathbf{H})^{\mathsf{T}}(\mathbf{H}'-\mathbf{H}) \\ &= \mathbf{H}^{\mathsf{T}}\mathbf{H}'-\mathbf{H}^{\mathsf{T}}\mathbf{H}+\mathbf{H}'^{\mathsf{T}}\mathbf{H}-\mathbf{H}^{\mathsf{T}}\mathbf{H}+\mathbf{H}'^{\mathsf{T}}\mathbf{H}'-\mathbf{H}'^{\mathsf{T}}\mathbf{H}-\mathbf{H}^{\mathsf{T}}\mathbf{H}'+\mathbf{H}^{\mathsf{T}}\mathbf{H} \\ &= \mathbf{H}'^{\mathsf{T}}\mathbf{H}'-\mathbf{H}^{\mathsf{T}}\mathbf{H}. \end{aligned} \tag{3.10}$$

□

Analisando (3.9), pode-se notar que o uso de uma matriz de afinidade no treinamento da ELM leva a um efeito de regularização de Tikhonov (Tikhonov, 1963). A matriz Λ representa o termo de regularização e é calculada usando apenas $\mathbf{H}$ e $\mathbf{P}$. Caso uma matriz de afinidade sem parâmetros seja utilizada, como a matriz de similaridade de cossenos (2.41), não é necessário ajustar parâmetro de regularização algum.

Além disso, é importante notar que (3.9) utiliza $\mathbf{Y}' = \mathbf{P}\mathbf{Y}$ em vez de $\mathbf{Y}$. Ou seja, para calcular os pesos da camada de saída, cada rótulo y_i é transformado do seu valor original em uma soma ponderada de todos os demais rótulos, $y'_i = \sum_{k=1}^{N} p_{ik}y_k$, na qual o peso associado com um dado y_k é definido como a similaridade entre $\mathbf{x}_i$ e $\mathbf{x}_k$. Essa modificação pode incrementar a generalização da ELM para problemas de aprendizado semissupervisionado, pois ao usar $\mathbf{Y}'$ no lugar de $\mathbf{Y}$, a informação *a priori* expressa por $\mathbf{P}$ é usada para definir os rótulos para padrões não-rotulados. Ou seja, a saída para cada padrão não-rotulado é definida como uma soma ponderada dos rótulos de todos os padrões rotulados.

Após o treinamento da rede, a mesma pode ser submetida a padrões desconhecidos - padrões de teste. Nesse caso, para cada padrão de teste, é calculado seu valor de afinidade com cada um dos padrões de treinamento e, após a projeção do padrão de teste na camada escondida, é

feito o cálculo de $\mathbf{h}'$, o qual é a projeção perturbada desse padrão. O Algoritmo 2 apresenta a projeção de um padrão de teste.

Algoritmo 2: Projeção de um padrão de teste.

Entrada: **H**: saída da camada escondida para o padrão de teste
XTrain: padrões de treinamento
xTest: padrão de teste a ser projetado
Z: matriz de pesos da camada escondida
Saída: $\mathbf{h}'$: projeção do padrão de teste, já modificada pela afinidade.

$\mathbf{p} \leftarrow \texttt{afinidade}([\mathbf{XTrain}; \mathbf{xTest}])$ {Calcula a afinidade da matriz formada pelos padrões de treinamento e padrão de teste}
$\mathbf{h} \leftarrow \texttt{funcaoAtivacao}(\mathbf{xTest} * \mathbf{Z})$;
$\mathbf{H}^* \leftarrow [\mathbf{h}; \mathbf{H}]$;
$\mathbf{h}' \leftarrow \mathbf{p} * \mathbf{H}^*$;

3.3 COMPLEXIDADE COMPUTACIONAL

O custo computacional da ELM é dominado pelo cálculo da pseudo-inversa, o qual, por sua vez, é dominado pelo custo da computação da Decomposição em Valores Singulares - *Singular Value Decomposition* (SVD). Segundo (Golub e Van Loan, 1996), o custo de calcular a SVD de uma matriz $N \times \ell$ é $O(4N\ell^2 + 22\ell^3)$. O método AMR-ELM possui um custo computacional maior que a ELM original, pois é necessário calcular a matriz de afinidade **P** e, depois, fazer a multiplicação de **P** por **H**. No caso da similaridade de cossenos, o custo de computar a matriz de afinidade é é $O(mN^2)$, e o custo de calcular o produto **PH** é $O(N^2\ell)$, no qual N é o número de padrões, m é o número de atributos e ℓ é o número de neurônios na camada escondida. Assim, o custo total adicionado é $O(N^2(\ell + m))$. Essa adição de custo é um compromisso (*tradeoff*), tendo em vista a melhora de desempenho obtida com a regularização.

3.4 CONCLUSÃO DO CAPÍTULO

Neste capítulo foi apresentado o método desenvolvido, AMR-ELM, o qual possibilita utilizar matrizes de afinidade para prover efeito de regularização de Tikhonov para a ELM. Além disso, por estimar rótulos ausentes por meio de uma soma, ponderada pelos valores de afinidade entre os padrões, dos rótulos de todos os padrões rotulados, o método possibilita que a ELM seja capaz de lidar com aprendizado semissupervisionado. Foram apresentados ainda os algoritmos de projeção de padrões de treinamento e de teste, além da análise de complexidade computacional do método, em comparação com a ELM original.

4

EXPERIMENTOS E RESULTADOS

Neste capítulo, o desempenho do método desenvolvido, AMR-ELM é comparado com a ELM original e com duas versões melhoradas da ELM: uma versão, chamada neste texto de ELM-Reg, na qual é acrescentado um termo de regularização no cálculo da pseudoinversa, conforme apresentado em (Huang et al., 2012), e a OP-ELM (Miche et al., 2008, 2010b). Dois cenários são utilizados: aprendizado supervisionado e aprendizado semissupervisionado. Com relação ao primeiro cenário, a comparação é feita com o objetivo de mostrar experimentalmente o ganho de desempenho de generalização provocado pela regularização, especialmente em comparação com a ELM. No que diz respeito ao segundo cenário, objetiva-se mostrar que a informação *a priori* inserida pelas matrizes de afinidade permite ao classificador obter um bom desempenho ainda que a quantidade de padrões rotulados represente apenas uma pequena fração dos padrões de treinamento. Foram utilizados dados sintéticos para mostrar as superfícies de separação geradas pelo método, comparando-as com a ELM, e bases de dados reais obtidas no repositório UCI (Asuncion e Newman, 2007) para comparar a acurácia de classificação da AMR-ELM com os demais classificadores.

4.1 METODOLOGIA

O procedimento utilizado para a realização dos experimentos é apresentado nos parágrafos a seguir.

Primeiramente, os dados obtidos no repositório UCI foram *pré-processados*: todos os padrões com valores ausentes foram removidos e, após isso, os dados foram normalizados para média 0 e desvio-padrão 1. Os rótulos foram ajustados para estarem em $\{-1, 1\}$. Posteriormente, fez-se a configuração geral, selecionando-se *função de ativação*, *matriz de afinidade*, *intervalo de amostragem dos pesos da camada escondida* e *número de neurônios* nessa camada. A *função de ativação* utilizada em todos os experimentos foi a sigmóide. Os *pesos da camada escondida* foram amostrados, de uma distribuição uniforme, no intervalo $[-0.5\ \ 0.5]$. A *matriz de afinidade* utilizada foi a matriz de similaridade de cossenos (2.41). Para a ELM-Reg, o parâmetro de regularização C é selecionado no intervalo $[2^{-24}, 2^{25}]$, conforme sugerido em (Huang et al., 2012), utilizando 10-*fold cross-validation*.

O *número de neurônios na camada escondida* (ℓ), no cenário do aprendizado supervisionado, varia em $\{10, 30, 100, 500, 1000\}$. É importante destacar que, pelo fato de a OP-ELM ser um método de poda, os números de neurônios na camada escondida são os valores iniciais, sendo

limitados pelo número de padrões de treinamento. Já no cenário semissupervisionado, ℓ foi fixado em 100. Ainda neste último cenário, a *porcentagem de padrões rotulados* nl varia em $NL = \{5\%, 10\%, 20\%, 50\%\}$. Para cada valor nl_i do vetor NL, são selecionados $nl_i * N$ padrões para terem seus rótulos preservados, onde N é o número de padrões de treinamento. Os demais rótulos recebem 0.

Finalizada a configuração, a rede então é treinada: os padrões de treinamento são projetados no espaço de caraterísticas ELM - Equação 2.18 e Algoritmo (1) - e a matriz de pesos da camada de saída, $\mathbf{W}$, é calculada obedecendo a Equação (2.21). Após o treinamento, a rede é testada, ou seja, são apresentados padrões que não foram apresentados durante o treinamento. Esse procedimento é o mesmo, tanto para o aprendizado supervisionado quanto para o semissupervisionado. Após a projeção, é calculada a saída da rede, $\hat{\mathbf{Y}}_{test}$ (Equação 2.25) e, por fim, feito o cálculo da acurácia de classificação.O procedimento de treinamento e teste foi executado com 10-*fold cross-validation* por 3 vezes, totalizando 30 execuções. Após isso, foram calculadas as médias da acurácia de teste e, para o cenário supervisionado, do tempo de treinamento.

4.1.1 *Testes estatísticos*

Comparar diferentes classificadores pela simples observação do desempenho de classificação em diversas bases de dados é uma tarefa difícil, tendo em vista que os resultados variam de acordo com a base. Dessa forma, (Demšar, 2006) e (Japkowicz e Shah, 2011) apresentam testes estatísticos que podem ser utilizados para essa comparação, com destaque para o teste de Friedman (Friedman, 1937, 1940), por ser não-paramétrico. No teste de Friedman, os algoritmos são "ranqueados" em cada base separadamente: o melhor recebe 1, o segundo melhor recebe 2 e assim por diante - em caso de empate, são atribuídos *ranks* médios. Para cada classificador j, a soma R_j dos *ranks* obtidas em todas as bases é calculada, e o resultado do teste é dado por (Japkowicz e Shah, 2011):

$$\chi_F^2 = \left[\frac{12}{n \times k \times (k+1)} \sum_{j=1}^{k} (R_j)^2 \right] - 3 \times n \times (k+1), \tag{4.1}$$

com n representando o número de bases de dados e k o número de classificadores.

Dado o valor de χ_F^2, consulta-se a tabela de valores críticos, disponível em (Japkowicz e Shah, 2011), de acordo com o nível de significância desejado. Caso χ_F^2 seja maior que o valor encontrado na tabela, pode-se rejeitar a hipótese nula (H_0) de equivalência entre os classificadores. Neste trabalho, foi calculado diretamente o p-valor com o auxílio do software R (R Core Team, 2013): caso o p-valor encontrado seja menor

que o grau de significância desejado, então há diferença significativa entre os classificadores.

A existência de diferença significativa entre os classificadores não diz, por exemplo, se todos são diferentes ou se apenas um deles tem desempenho significativamente diferente enquanto os demais podem ser considerados estatisticamente iguais para aquelas bases de dados. Para resolver esse problema, (Demšar, 2006) e (Japkowicz e Shah, 2011) sugerem a realização de um teste *post hoc*, o qual pode ser o teste de Nemenyi (Nemenyi, 1963), aplicado para comparar os algoritmos "um-contra-um", indicando onde estão as diferenças significativas. Nesse teste, se os *ranks* médios de dois algoritmos diferem de pelo menos (Castro, 2011)

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6n}}, \tag{4.2}$$

com q_α sendo encontrado na tabela de valores críticos para o teste de Nemenyi, disponível em (Demšar, 2006), de acordo com o nível de significância α.

Em (Demšar, 2006) é apresentado um diagrama que permite uma visualização interessante dessa estatística, o Diagrama de Diferença Crítica - *Critical Difference Diagram* (CDD). A Figura 4.1 mostra um exemplo de CDD, com 4 classificadores hipotéticos: A, B, C e D. O valor de diferença crítica (CD) é apresentado como uma linha no topo do gráfico. O eixo apresenta os *ranks* médios, ordenados da direita para a esquerda - o melhor aparece à direita. A identificação do classificador aparece ao lado da respectiva linha, abaixo do eixo, enquanto o *rank* médio do algoritmo aparece sobre essa mesma linha. Os grupos conectados não são significativamente diferentes uns dos outros, ao nível α de significância. Nesse caso, o classificador C tem o melhor *rank* médio $(1,15)$, mas é estatisticamente equivalente ao classificador A. Por outro lado, C é significativamente melhor que B e D, pois não há linha conectando-os. Os CDD apresentados neste trabalho foram desenvolvidos utilizando o software MatLab (MATLAB, 2009).

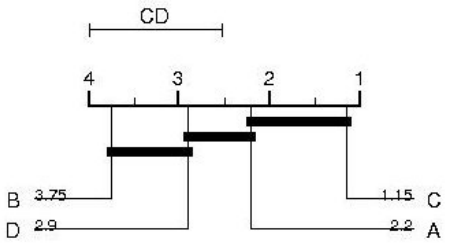


Figura 4.1: Exemplo de CDD. Linha acima do eixo representa o valor de Diferença Crítica (CD). Eixo representa *ranks* médios dos algoritmos, com o melhor à direita. Grupos conectados não são significativamente diferentes.

4.2 BASES DE DADOS

4.2.1 *Dados Sintéticos*

Os dados sintéticos utilizados nos experimentos iniciais são apresentados na Figura 4.2. Cada agrupamento, contendo 50 pontos, foi gerado a partir de uma distribuição normal bivariada, com médias $[-2 \ -2]$ e $[2 \ 2]$. Para o cenário semissupervisionado, foram escolhidos 10 padrões de cada classe para serem rotulados. A geração desses dados tem como objetivo permitir realizar os experimentos de forma controlada, observando visualmente os resultados obtidos e a capacidade do método quando os padrões possuem baixa dimensionalidade.

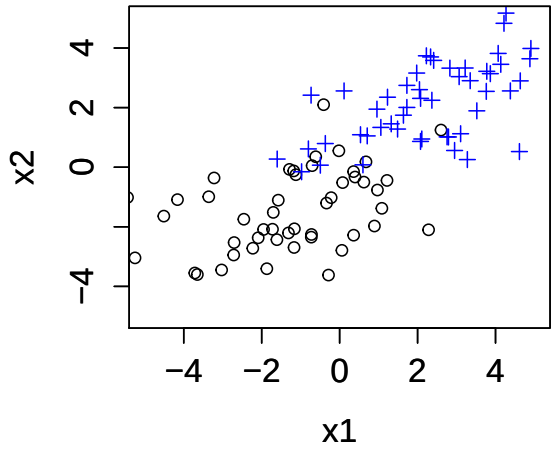


Figura 4.2: Dados sintéticos.

4.2.2 *Dados Reais*

A Tabela 4.1 apresenta os nomes das bases reais utilizadas nos experimentos, assim como os acrônimos que serão utilizados para referenciá-las neste texto. Todas as bases correspondem a problemas de classificação binária ou foram adaptadas para isso - caso específico da base `thy`-, e foram obtidas no conhecido repositório da UCI (*University of California, Irvine*) (Asuncion e Newman, 2007). As características apresentadas na Tabela 4.2 já levam em consideração a remoção de atributos com valores ausentes, realizada no pré-processamento.

A seguir é apresentada uma breve descrição das bases, seguida da Tabela 4.2, que apresenta as principais características (número de padrões e número de atributos) das mesmas.

Tabela 4.1: Nomes das bases reais e seus respectivos acrônimos.

Nome	**Acrônimo**
Australian Credit	`acr`
QSAR biodegradation	`bio`
BUPA Liver Disorders	`bld`
Statlog (Heart)	`hea`
Mushroom	`mshr`
Pima Indians Diabetes	`pid`
Splice-junction Gene Sequences	`spl`
Thyroid disease	`thy`
Congressional Voting Records	`vot`
Wisconsin Breast Cancer	`wbc`

- `acr`: base sobre concessão de crédito na Austrália, com o objetivo de classificar entre passível de receber crédito ou não.
- `bio`: descritores moleculares de químicos, para classificação entre biodegradável e não-biodegradável.
- `bld`: base de dados da *Bupa Medical Research Ltd.* contendo testes sanguíneos de homens. O objetivo é identificar se o paciente é portador de desordens no fígado.
- `hea`: dados de imagens cardíacas obtidas por tomografia computadorizada. Cada paciente é classificado como normal ou anormal.
- `mshr`: descrição de cogumelos em termos de características físicas, classificados em venenosos ou comestíveis.
- `pid`: dados de mulheres de um povo indígena norte-americano (*Pima*), cuja incidência de diabetes é maior que no restante da população. Classificação em portador ou não de diabetes.
- `spl`: classificação de pontos de junção de DNA.
- `thy`: classificação entre portador ou não de problemas na tireóide (as classes hipo e hiper tireoidismo foram agrupadas como "portador").
- `vot`: Votos para cada um dos deputados dos Estados Unidos no ano de 1984. De acordo com os 16 votos em temas considerados importantes, deve-se classificar cada voto entre republicano e democrata.
- `wbc`: Classificação de tumores de mama como benignos ou malignos.

Tabela 4.2: Principais características das bases reais.

Base	# Padrões	# Atributos
acr	690	14
bio	1055	41
bld	345	6
hea	270	13
mshr	8124	21
pid	768	8
spl	3190	60
thy	215	5
vot	435	16
wbc	683	10

4.3 RESULTADOS DE CLASSIFICAÇÃO E TESTES ESTATÍSTICOS

4.3.1 *Aprendizado Supervisionado*

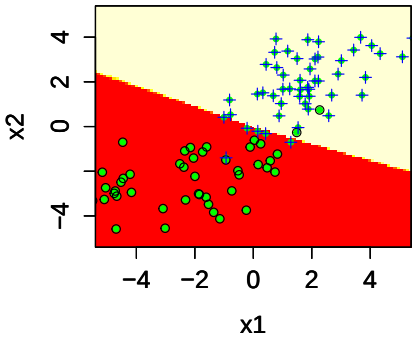
Com o objetivo de avaliar o efeito de regularização observado pela inclusão da informação *a priori* por meio de matrizes de afinidade durante o treinamento da ELM, nesta subseção são apresentadas as superfícies de separação médias obtidas, para os dados sintéticos, pelo método desenvolvido (AMR-ELM), utilizando matriz de similaridade de cossenos, e pela ELM, assim como os resultados de classificação (acurácia de teste) para as bases de dados reais descritas anteriormente, comparando a AMR-ELM, novamente utilizando matriz de similaridade de cossenos, com a ELM original, com a ELM Regularizada (ELM-Reg) e com a OP-ELM. Também são apresentados os tempos de treinamento.

A Figura 4.3 apresenta as superfícies de separação médias para a AMR-ELM e para a ELM, com diferentes valores de neurônios na camada escondida (ℓ).

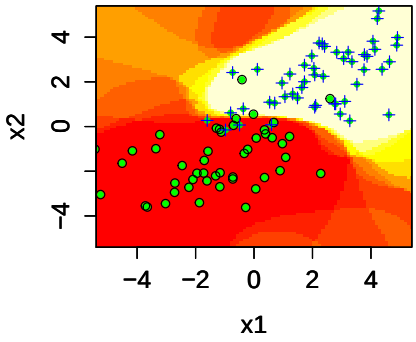
A Tabela 4.3 detalha a acurácia de teste (média $\pm$ desvio-padrão) para diferentes números de neurônios na camada escondida (ℓ), para cada base. Os melhores resultados estão em negrito, assim como os resultados que diferem em menos de 1% dos melhores. Os mesmos resultados são apresentados sob a forma de gráficos, nas Figuras A.1 - A.9 do Apêndice A.

A Tabela 4.4 apresenta os tempos de treinamento médio, em segundos, para os diferentes números de neurônios na camada escondida (ℓ), de acordo com a base de dados.

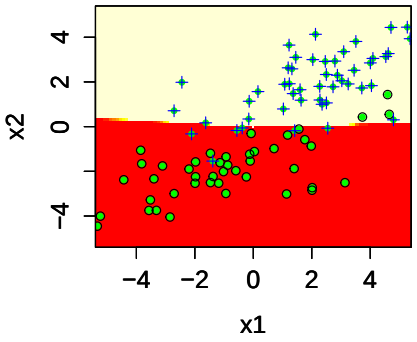
A Tabela 4.5 apresenta os p-valores resultantes do teste de Friedman, para cada valor de neurônios na camada escondida (ℓ). Os p-valores menores que 0.05 são apresentados em negrito. Para possibilitar a exe-



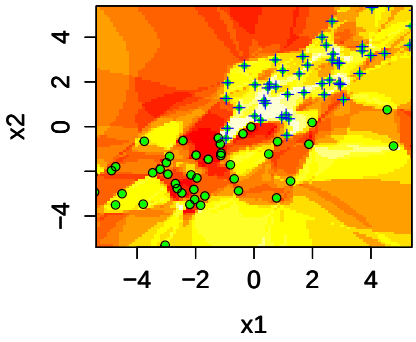
(a) AMR-ELM - $\ell = 20$



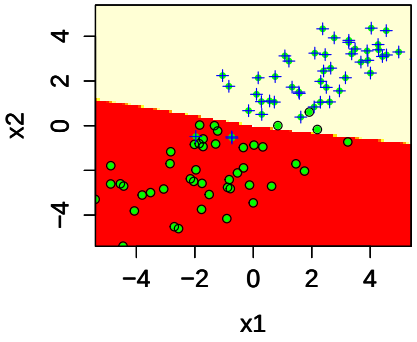
(b) ELM - $\ell = 20$



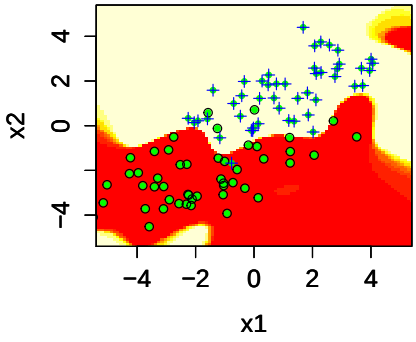
(c) AMR-ELM - $\ell = 100$



(d) ELM - $\ell = 100$



(e) AMR-ELM - $\ell = 500$



(f) ELM - $\ell = 500$

Figura 4.3: Superfícies de separação médias para diferentes valores de ℓ, para o cenário supervisionado. Cada uma das duas classes possui 50 pontos.

Tabela 4.3: Acurácia de teste (*média* ± *desvio-padrão*) para aprendizado supervisionado. Os resultados são exibidos de acordo com o método, o valor de ℓ e a base. Os melhores resultados - assim como aqueles que diferem em menos de 1% dos melhores - são apresentados em negrito.

ℓ	**AMR-ELM**	**ELM**	**ELM-Reg**	**OP-ELM**
		acr		
10	**85.65 ± 4.2**	82.08 ± 4.8	81.84 ± 5.8	77.93 ± 6.6
30	**86.28 ± 4.6**	**86.52 ± 4.6**	**86.57 ± 4.3**	84.88 ± 5.6
100	**86.57 ± 4.2**	84.93 ± 4.7	**86.09 ± 5.3**	85.64 ± 2.8
500	**86.47 ± 3.4**	66.81 ± 5.0	**86.86 ± 3.8**	82.76 ± 4.4
1000	**86.47 ± 5.2**	50.39 ± 5.6	**86.52 ± 5.1**	-
		bio		
10	71.35 ± 3.8	70.17 ± 5.4	**72.41 ± 4.3**	70.87 ± 4.4
30	74.06 ± 5.2	**79.62 ± 3.7**	**80.06 ± 4.3**	77.15 ± 4.9
100	**83.16 ± 3.9**	**82.98 ± 4.4**	**83.01 ± 4.3**	80.76 ± 3.5
500	**83.89 ± 3.5**	74.50 ± 4.1	**83.99 ± 4.0**	79.49 ± 3.4
1000	83.51 ± 3.9	55.07 ± 5.2	**85.22 ± 3.9**	-
		bld		
10	65.53 ± 8.2	**68.09 ± 7.0**	66.90 ± 7.3	60.61 ± 8.5
30	64.94 ± 7.5	**69.79 ± 7.0**	**68.42 ± 6.7**	64.94 ± 7.2
100	65.35 ± 8.1	67.86 ± 8.5	**71.52 ± 6.5**	67.12 ± 7.7
500	65.99 ± 6.6	52.02 ± 7.6	**70.46 ± 7.0**	-
1000	65.71 ± 8.7	54.41 ± 9.2	**71.49 ± 8.1**	-
		hea		
10	**84.44 ± 6.2**	79.38 ± 7.8	80.74 ± 7.4	77.41 ± 8.7
30	**83.95 ± 8.1**	**84.44 ± 8.4**	**84.07 ± 7.3**	80.00 ± 6.9
100	**84.20 ± 5.7**	74.20 ± 6.8	**84.44 ± 6.3**	80.37 ± 6.6
500	**83.83 ± 7.2**	69.14 ± 7.3	**84.32 ± 7.4**	-
1000	**83.95 ± 6.5**	71.73 ± 9.4	**84.32 ± 7.3**	-
		mshr		
10	**92.12 ± 1.5**	89.70 ± 3.4	89.20 ± 3.1	85.33 ± 3.6
30	**96.04 ± 0.7**	**96.42 ± 0.9**	**96.47 ± 1.0**	93.00 ± 1.7
100	96.01 ± 0.7	**99.61 ± 0.2**	**99.65 ± 0.3**	98.05 ± 0.7
500	96.07 ± 0.6	**100.00 ± 0.0**	**100.00 ± 0.0**	**99.95 ± 0.1**
1000	96.06 ± 0.7	**100.00 ± 0.0**	**100.00 ± 0.0**	**99.98 ± 0.1**
		pid		
10	72.84 ± 5.0	**76.30 ± 4.7**	**76.43 ± 4.6**	72.52 ± 4.8
30	74.09 ± 4.5	**77.00 ± 5.2**	**76.39 ± 5.0**	75.13 ± 4.9
100	74.39 ± 4.4	74.46 ± 5.4	**77.20 ± 4.1**	74.79 ± 5.8
500	74.08 ± 5.3	61.68 ± 5.2	**76.91 ± 3.0**	70.70 ± 4.6
1000	73.87 ± 4.3	58.28 ± 5.9	**76.95 ± 5.2**	-
		spl		
10	**76.43 ± 3.8**	64.70 ± 4.2	66.18 ± 3.6	62.99 ± 4.0
30	**84.59 ± 2.4**	76.88 ± 4.3	75.95 ± 3.4	72.06 ± 3.4
100	**88.83 ± 2.3**	87.30 ± 2.1	87.07 ± 2.2	82.47 ± 2.4
500	**88.74 ± 1.9**	86.92 ± 2.1	**89.04 ± 1.9**	87.21 ± 1.8
1000	**88.81 ± 1.5**	83.96 ± 2.1	**89.05 ± 1.6**	86.34 ± 2.1
		thy		
10	**90.34 ± 7.0**	89.10 ± 7.6	87.99 ± 7.4	**89.48 ± 6.4**
30	89.70 ± 7.2	92.97 ± 6.7	90.93 ± 8.4	**94.71 ± 4.9**
100	90.51 ± 6.5	85.84 ± 6.9	**94.25 ± 6.3**	**94.91 ± 4.7**
500	89.48 ± 5.4	85.25 ± 7.2	**96.26 ± 4.7**	-
1000	89.29 ± 6.3	85.82 ± 9.6	**95.51 ± 5.1**	-
		vot		
10	**90.32 ± 4.3**	**90.87 ± 3.9**	89.42 ± 5.3	87.27 ± 4.7
30	**94.03 ± 3.9**	**93.25 ± 3.7**	**93.65 ± 4.1**	91.25 ± 4.2
100	**94.25 ± 3.3**	92.87 ± 3.3	**93.86 ± 3.5**	92.57 ± 4.4
500	**94.49 ± 3.8**	85.82 ± 5.4	**94.04 ± 3.7**	-
1000	**94.33 ± 3.5**	88.66 ± 4.6	**94.56 ± 3.9**	-
		wbc		
10	**96.09 ± 5.9**	**96.48 ± 2.2**	**96.29 ± 1.9**	95.17 ± 2.2
30	**97.22 ± 1.6**	**96.59 ± 2.2**	**96.24 ± 2.2**	**96.34 ± 2.3**
100	**97.27 ± 1.6**	96.05 ± 2.6	**96.58 ± 1.6**	**97.07 ± 1.8**
500	**97.41 ± 1.6**	78.33 ± 3.3	**96.97 ± 2.1**	96.04 ± 2.2
1000	**97.22 ± 1.7**	85.11 ± 3.3	**96.87 ± 2.0**	-

Tabela 4.4: Tempos médios de treinamento, em segundos, de acordo com o método, o valor de ℓ e a base, para o aprendizado supervisionado.

ℓ	**AMR-ELM**	**ELM**	**ELM-Reg**	**OP-ELM**
		acr		
10	0.03	0.004	1.94	0.04
30	0.04	0.009	3.76	0.06
100	0.10	0.05	15.95	0.50
500	1.21	1.44	370.44	56.72
1000	3.07	3.51	781.99	-
		bio		
10	0.15	0.005	653.16	0.04
30	0.12	0.02	671.65	0.10
100	0.24	0.07	746.51	0.66
500	2.55	2.44	1170.05	59.04
1000	6.56	7.61	1712.10	-
		bld		
10	0.01	0.002	1.090	0.02
30	0.01	0.005	1.89	0.04
100	0.04	0.03	8.51	0.33
500	0.52	0.52	101.98	-
1000	0.57	0.79	167.47	-
		hea		
10	0.006	0.002	0.91	0.01
30	0.01	0.005	1.69	0.03
100	0.03	0.03	7.13	0.27
500	0.27	0.30	63.42	-
1000	0.33	0.47	109.24	-
		mshr		
10	4.64	0.11	18.83	0.19
30	6.41	0.08	37.31	0.70
100	12.60	0.43	158.06	5.07
500	51.36	9.02	2848.39	143.23
1000	109.98	38.65	11877.73	1188.15
		pid		
10	0.03	0.0030	1.98	0.02
30	0.10	0.0080	3.86	0.06
100	0.16	0.05	16.56	0.51
500	1.59	1.60	383.28	55.27
1000	3.79	4.31	866.16	-
		spl		
10	1.26	0.02	9.86	0.18
30	1.56	0.04	18.97	0.39
100	2.63	0.19	73.11	2.43
500	10.23	4.17	1234.75	100.73
1000	25.45	20.54	5360.76	1037.19
		thy		
10	0.004	0.001	0.69	0.01
30	0.006	0.003	1.19	0.03
100	0.03	0.03	5.41	0.25
500	0.13	0.15	33.82	-
1000	0.20	0.25	63.83	-
		vot		
10	0.02	0.004	1.25	0.04
30	0.02	0.005	2.36	0.06
100	0.10	0.04	10.91	0.41
500	0.70	0.79	162.62	-
1000	0.87	1.13	260.61	-
		wbc		
10	0.02	0.003	1.75	0.02
30	0.09	0.007	3.34	0.06
100	0.14	0.05	14.60	0.46
500	1.29	1.43	350.97	55.04
1000	3.06	3.18	649.51	-

cução dos testes, os valores ausentes da OP-ELM foram substituídos pela média dos demais valores com ℓ menor.

Tabela 4.5: Resultados dos Testes de Friedman para o aprendizado supervisionado: p-valores para cada ℓ. Os p-valores menores que 0.05 estão em negrito.

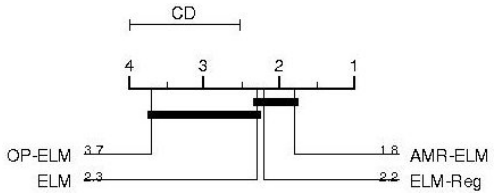
ℓ	**p-valor**
10	**0.006246401**
30	**0.03360332**
100	0.08580109
500	**0.0001115577**
1000	**0.0000642909**

De acordo com a Tabela 4.5, a hipótese nula de equivalência entre os métodos (H_0) pode ser rejeitada a um nível de significância de 5% para $\ell \in \{10, 30, 500, 1000\}$. De forma a comparar os métodos um contra o outro, as Figuras 4.4a - 4.4d apresentam os CDDs (Demšar, 2006) para esses valores de ℓ. O valor de diferença crítica é apresentado como uma linha sobre o gráfico, enquanto o eixo apresenta os *ranks* médios - o melhor aparece à direita. Grupos conectados não são significativamente diferentes a $p = 0.05$.

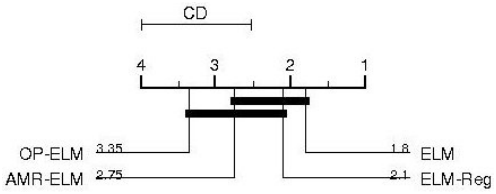
Bartlett (1998) afirma que o tamanho dos pesos de uma RNA é mais importante do que o tamanho da rede (número de parâmetros) para uma boa capacidade de generalização, ou seja, uma rede com muitos neurônios pode possuir boa capacidade de generalização caso os a magnitude dos seus pesos seja mantida pequena. Com o objetivo de comparar as normas dos pesos do método AMR-ELM com as normas dos pesos da ELM original, a Tabela 4.6 apresenta os valores da norma dos pesos da camada de saída ($\|\mathbf{W}\|$) da AMR-ELM e da ELM original (média $\pm$ desvio-padrão), para o aprendizado supervisionado, de acordo com os diferentes valores de ℓ.

4.3.2 *Aprendizado Semissupervisionado*

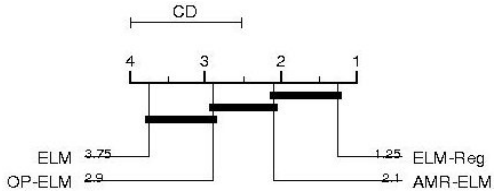
Conforme visto no Capítulo 3, no método desenvolvido, o valor de cada rótulo y_i é transformado do seu valor original em uma soma ponderada de todos os demais rótulos, o que pode incrementar a generalização da ELM para problemas de aprendizado semissupervisionado. Assim, nesta Seção, o desempenho do método é avaliado usando problemas semissupervisionados. Como todas as bases de dados utilizadas neste trabalho são originalmente para problemas de classificação supervisionada, elas tiveram que ser modificadas conforme descrito na Seção 4.1, exceto para a OP-ELM, para a qual os padrões com rótulos removidos foram descartados.



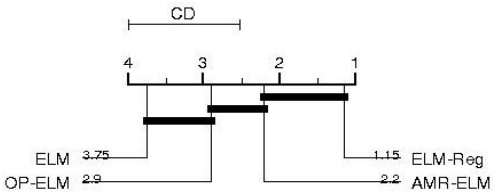
(a) CDD para $\ell = 10$.



(b) CDD para $\ell = 30$.



(c) CDD para $\ell = 500$.



(d) CDD para $\ell = 1000$.

Figura 4.4: CDDs para o aprendizado supervisionado. O valor de diferença crítica é apresentado como uma linha sobre o gráfico. O eixo apresenta os *ranks* médios, com o melhor à direita. Grupos conectados não são significativamente diferentes a $p = 0.05$.

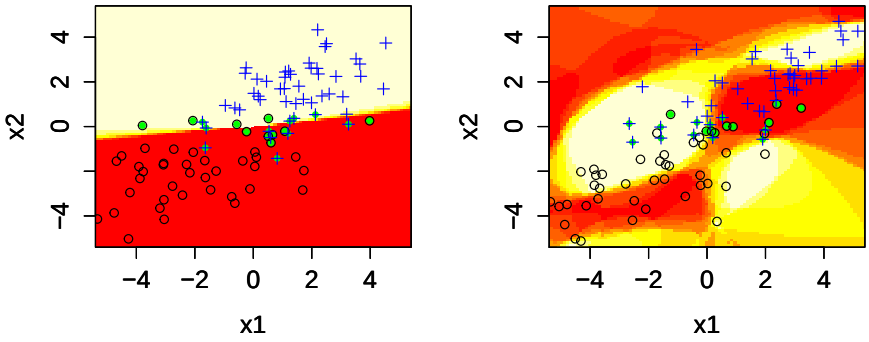
Tabela 4.6: Normas dos pesos da camada de saída (*média* ± *desvio-padrão*) para a AMR-ELM e para a ELM, de acordo com o valor de ℓ - cenário supervisionado.

ℓ	**AMR-ELM**	**ELM**	ℓ	**AMR-ELM**	**ELM**
	acr			**pid**	
10	0.11 ± 0.0	3.76 ± 0.0	10	0.06 ± 0.0	6.42 ± 0.0
30	0.09 ± 0.0	5.42 ± 0.0	30	0.02 ± 0.0	15.86 ± 0.0
100	0.04 ± 0.0	15.72 ± 0.0	100	0.01 ± 0.0	59.65 ± 0.0
500	0.02 ± 0.0	371.28 ± 0.0	500	0.00 ± 0.0	5040.62 ± 0.0
1000	0.01 ± 0.0	3514.22 ± 0.0	1000	0.00 ± 0.0	9042.73 ± 0.0
	bio			**spl**	
10	0.05 ± 0.0	1.47 ± 0.0	10	0.08 ± 0.0	1.03 ± 0.0
30	0.45 ± 0.2	2.39 ± 0.2	30	0.14 ± 0.0	1.79 ± 0.0
100	0.87 ± 0.1	4.33 ± 0.1	100	0.13 ± 0.0	2.66 ± 0.0
500	0.31 ± 0.0	21.29 ± 0.0	500	0.04 ± 0.0	8.67 ± 0.0
1000	0.22 ± 0.0	241.45 ± 0.0	1000	0.03 ± 0.0	20.75 ± 0.0
	bld			**thy**	
10	0.43 ± 0.1	13.31 ± 0.1	10	0.20 ± 0.1	44.85 ± 0.1
30	0.16 ± 0.0	51.56 ± 0.0	30	0.09 ± 0.0	278.55 ± 0.0
100	0.09 ± 0.0	893.79 ± 0.0	100	0.04 ± 0.0	29062.89 ± 0.0
500	0.04 ± 0.0	112764.19 ± 0.0	500	0.02 ± 0.0	19927.14 ± 0.0
1000	0.03 ± 0.0	28388.28 ± 0.0	1000	0.01 ± 0.0	12595.69 ± 0.0
	hea			**vot**	
10	0.22 ± 0.1	3.89 ± 0.1	10	0.23 ± 0.1	3.39 ± 0.1
30	0.11 ± 0.0	9.12 ± 0.0	30	0.23 ± 0.1	6.09 ± 0.1
100	0.05 ± 0.0	32.62 ± 0.0	100	0.10 ± 0.0	14.81 ± 0.0
500	0.02 ± 0.0	63.62 ± 0.0	500	0.04 ± 0.0	46.15 ± 0.0
1000	0.01 ± 0.0	36.20 ± 0.0	1000	0.03 ± 0.0	23.02 ± 0.0
	mshr			**wbc**	
10	0.02 ± 0.0	3.47 ± 0.0	10	0.73 ± 1.6	4.38 ± 1.6
30	0.11 ± 0.0	7.00 ± 0.0	30	0.03 ± 0.0	9.50 ± 0.0
100	0.03 ± 0.0	12.66 ± 0.0	100	0.01 ± 0.0	33.82 ± 0.0
500	0.01 ± 0.0	18.08 ± 0.0	500	0.01 ± 0.0	9710.41 ± 0.0
1000	0.01 ± 0.0	20.07 ± 0.0	1000	0.00 ± 0.0	438.83 ± 0.0

A Figura 4.5 apresenta as superfícies de separação médias para a AMR-ELM e para a ELM, com diferentes valores de neurônios na camada escondida (ℓ). Foram selecionados 10 padrões em cada classe para serem rotulados, e os demais permaneceram sem rótulos.

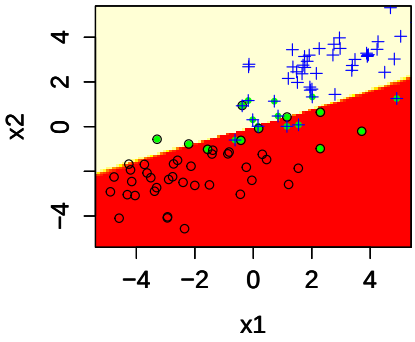
A Tabela 4.7 detalha a acurácia de teste (média $\pm$ desvio-padrão) para cada cenário considerado. O número de neurônios na camada escondida (ℓ) foi mantido em 100 em todos os experimentos. Valores ausentes para a OP-ELM indicam cenários nos quais o modelo não é capaz de executar apropriadamente. Os mesmos resultados são apresentados sob a forma de gráficos, nas Figuras A.10 - A.18 do Apêndice A.

De acordo com a Tabela 4.8, a hipótese nula de equivalência entre os métodos (H_0) pode ser rejeitada a um nível de significância de 5% para $\%nl \in \{0.1, 0.2, 0.5\}$. De forma a comparar os métodos um contra o outro, as Figuras 4.6a - 4.6c apresentam os CDDs (Demšar, 2006) para esses valores de $\%nl$. O valor de diferença crítica é apresentado como uma linha sobre o gráfico, enquanto o eixo apresenta os *ranks* médios - o melhor aparece à direita. Grupos conectados não são significativamente diferentes a $p = 0.05$.

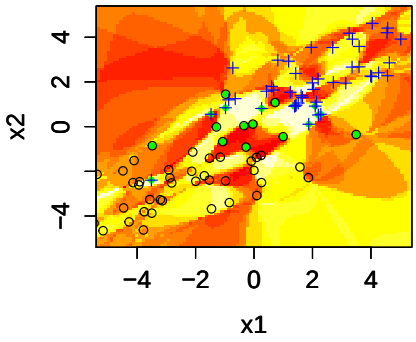


(a) AMR-ELM - $\ell = 20$

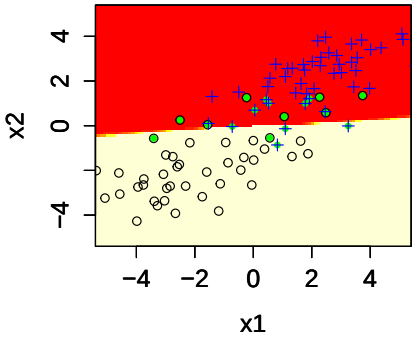
(b) ELM - $\ell = 20$



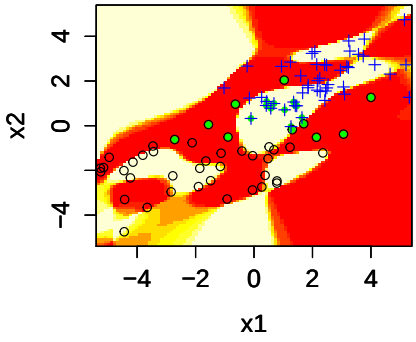
(c) AMR-ELM - $\ell = 100$



(d) ELM - $\ell = 100$



(e) AMR-ELM - $\ell = 500$



(f) ELM - $\ell = 500$

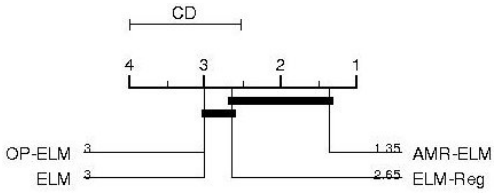
Figura 4.5: Superfícies de separação médias para diferentes valores de ℓ - cenário semissupervisionado, com 10 padrões rotulados, representados em verde, em cada uma das duas classes (total de 100 pontos).

Tabela 4.7: Acurácia de teste (*média* ± *desvio-padrão*) para aprendizado semissupervisionado. Os resultados são exibidos de acordo com o método, o valor de %nl e a base. Os melhores resultados - assim como aqueles que diferem em menos de 1% dos melhores - são apresentados em negrito.

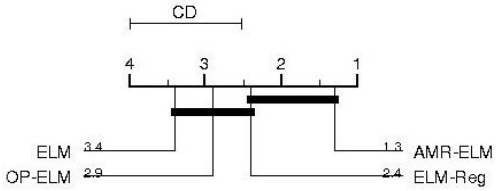
% Rotulados	**AMR-ELM**	**ELM**	**ELM-Reg**	**OP-ELM**
		acr		
0.05	**80.63 ± 4.3**	74.73 ± 5.2	64.49 ± 19.2	73.34 ± 8.0
0.1	**84.54 ± 4.9**	68.07 ± 10.5	**84.35 ± 5.5**	79.47 ± 6.4
0.2	**85.89 ± 4.4**	67.34 ± 8.8	**85.17 ± 5.5**	80.39 ± 5.8
0.5	**86.47 ± 3.8**	82.95 ± 5.1	**85.94 ± 3.9**	83.58 ± 4.0
		bio		
0.05	52.47 ± 6.6	**63.48 ± 6.0**	**64.56 ± 8.5**	-
0.1	**72.41 ± 5.9**	55.38 ± 6.1	70.90 ± 5.5	67.81 ± 5.2
0.2	**76.33 ± 5.0**	72.35 ± 4.7	72.64 ± 6.1	74.78 ± 4.3
0.5	79.87 ± 4.9	79.62 ± 3.7	**81.67 ± 3.7**	79.37 ± 3.7
		bld		
0.05	**59.28 ± 8.7**	**58.79 ± 9.7**	54.78 ± 12.1	45.20 ± 6.6
0.1	**60.90 ± 9.2**	53.15 ± 10.1	58.58 ± 9.0	43.86 ± 4.6
0.2	**64.48 ± 8.9**	53.83 ± 6.8	62.63 ± 8.8	50.22 ± 7.7
0.5	64.47 ± 9.5	60.99 ± 9.1	**67.63 ± 10.5**	**67.17 ± 7.4**
		hea		
0.05	65.93 ± 8.0	**70.25 ± 8.3**	60.00 ± 10.4	-
0.1	69.63 ± 9.5	66.67 ± 10.7	69.63 ± 17.4	**73.46 ± 9.4**
0.2	**79.01 ± 7.0**	70.99 ± 7.7	69.75 ± 14.9	76.54 ± 9.0
0.5	82.84 ± 6.9	64.32 ± 8.8	**83.95 ± 6.4**	78.89 ± 7.3
		mshr		
0.05	96.02 ± 1.0	**97.75 ± 0.8**	89.25 ± 2.6	96.51 ± 1.1
0.1	96.17 ± 0.6	**99.20 ± 0.4**	95.80 ± 2.1	97.26 ± 0.9
0.2	96.24 ± 0.6	**99.40 ± 0.3**	**98.66 ± 0.7**	**98.16 ± 0.6**
0.5	96.20 ± 0.6	**99.51 ± 0.3**	**99.44 ± 0.4**	97.90 ± 1.0
		pid		
0.05	65.53 ± 4.5	64.15 ± 6.0	**66.89 ± 4.4**	62.59 ± 5.7
0.1	**72.22 ± 5.0**	58.30 ± 6.3	70.15 ± 5.1	63.75 ± 6.9
0.2	**73.22 ± 4.5**	61.64 ± 6.8	**73.13 ± 5.4**	69.00 ± 7.1
0.5	72.74 ± 4.6	71.88 ± 4.0	76.31 ± 4.6	**74.71 ± 4.4**
		spl		
0.05	**75.59 ± 4.7**	58.61 ± 5.4	71.77 ± 9.2	71.51 ± 3.2
0.1	**84.61 ± 1.8**	80.84 ± 2.2	81.36 ± 3.4	77.50 ± 3.2
0.2	**87.55 ± 2.4**	84.10 ± 2.7	84.45 ± 2.7	79.61 ± 2.1
0.5	**88.20 ± 2.0**	86.17 ± 2.1	86.64 ± 1.9	81.78 ± 2.4
		thy		
0.05	**77.26 ± 15.7**	**77.27 ± 8.4**	70.26 ± 10.6	76.01 ± 11.9
0.1	**87.45 ± 6.7**	82.60 ± 8.9	77.05 ± 9.7	86.15 ± 7.4
0.2	**88.96 ± 6.1**	83.95 ± 7.8	85.58 ± 9.4	**88.10 ± 6.2**
0.5	88.69 ± 6.3	77.51 ± 9.4	88.97 ± 7.2	**92.46 ± 7.1**
		vot		
0.05	84.76 ± 5.0	**89.89 ± 3.6**	67.53 ± 12.4	75.42 ± 9.7
0.1	**88.82 ± 5.4**	**88.21 ± 5.4**	**87.92 ± 6.7**	85.54 ± 6.3
0.2	**92.18 ± 3.2**	85.50 ± 7.7	90.27 ± 6.0	86.21 ± 5.7
0.5	**94.25 ± 2.8**	90.18 ± 5.0	**93.26 ± 3.6**	89.91 ± 4.9
		wbc		
0.05	**95.42 ± 3.0**	91.61 ± 6.5	83.73 ± 16.7	62.61 ± 20.2
0.1	**96.69 ± 2.3**	92.44 ± 3.4	91.45 ± 9.8	64.65 ± 16.2
0.2	**96.68 ± 1.7**	83.22 ± 5.0	95.36 ± 2.9	85.99 ± 9.1
0.5	**97.31 ± 2.2**	92.58 ± 3.5	**96.73 ± 1.9**	96.28 ± 2.0

Tabela 4.8: Resultados dos Testes de Friedman para o aprendizado semisupervisionado: p-valores para cada ℓ. Os p-valores menores que 0.05 estão em negrito.

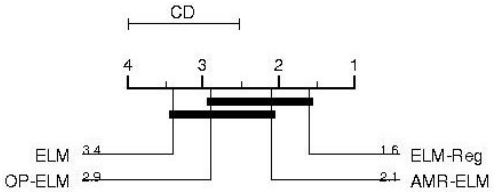
ℓ	**p-valor**
0.05	0.1604
0.1	**0.01078**
0.2	**0.002276**
0.5	**0.008724**



(a) CDD para %nl = 0.1.



(b) CDD para %nl = 0.2.



(c) CDD para %nl = 0.5.

Figura 4.6: CDDs para o aprendizado semisupervisionado. O valor de diferença crítica é apresentado como uma linha sobre o gráfico. O eixo apresenta os *ranks* médios, com o melhor à direita. Grupos conectados não são significativamente diferentes a $\mathrm{p} = 0.05$.

4.4 DISCUSSÕES

Os testes estatísticos do cenário supervisionado mostram que, quando o número de neurônios cresce ($\ell \in \{500, 1000\}$), o método AMR-ELM é estatisticamente equivalente à ELM-Reg e à OP-ELM, sendo superior à ELM original a um grau de significância de 5%. Isso confirma o efeito de suavização da resposta obtido pela regularização de Tikhonov do método, apresentado na Equação (3.9). Tal efeito de suavização pode, ainda, ser claramente observado nas superfícies de separação apresentadas na Figura 4.3: as superfícies obtidas pela AMR-ELM são mais suaves que as obtidas pela ELM. As normas dos pesos da camada de saída apresentadas na Tabela 4.6 também permitem observar o efeito de regularização, tendo em vista que, quando a norma dos pesos é pequena, a rede tende a ter uma melhor capacidade de generalização (Bartlett, 1998) e, para a AMR-ELM, as normas são mantidas em valores pequenos, não variando ordem de grandeza quando ℓ aumenta, enquanto na ELM original os valores tendem a ter um aumento significativo à medida que ℓ aumenta. Além disso, pelo fato de a AMR-ELM, ao ser configurada com uma afinidade não paramétrica como a similaridade de cossenos (2.41), não possuir parâmetros a serem ajustados, possui tempos de treinamento significativamente menores que os da ELM-Reg e menores que o da OP-ELM na grande maioria dos cenários (Tabela 4.4). Outra observação importante é a de que os resultados da AMR-ELM não apresentam grande variabilidade quando o número de neurônios aumenta (Tabela 4.3), ou seja, mesmo o número de neurônios não necessita de ser ajustado, bastando configurar a rede com um número relativamente grande de neurônios (por exemplo, $\ell \in \{500, 1000\}$) para se obter uma capacidade de generalização adequada.

Por outro lado, no cenário semissupervisionado, os testes estatísticos mostram que, com $\%\mathrm{NL} \in \{10\%, 20\%\}$, o método diferencia-se significativamente da ELM original, permitindo concluir que, em um cenário de rótulos escassos, o uso de matrizes de afinidade permite à ELM obter informação estrutural dos dados, melhorando sua capacidade de generalização nessa situação relativamente comum de escassez de rótulos. É interessante notar que, em alguns casos, com apenas 10% ou 20% dos rótulos disponíveis (Tabela 4.7), é possível obter resultados bastante próximos àqueles obtidos com todos os padrões rotulados (Tabela 4.3). Além disso, as superfícies de separação obtidas pelo método AMR-ELM, apresentadas na Figura 4.5, são consideravelmente mais suaves que aquelas obtidas pela ELM.

4.5 CONCLUSÃO DO CAPÍTULO

Neste capítulo, o método desenvolvido, AMR-ELM, foi comparado com outros métodos: ELM original e versões melhoradas da ELM. No cenário supervisionado, o método apresentou desempenho estatisticamente

equivalente à ELM-Reg e à OP-ELM para a maioria dos valores de ℓ, com a vantagem de possuir um tempo de treinamento significativamente menor que a ELM-Reg e, na maioria das situações, menor que a OP-ELM. Já no cenário semissupervisionado, foi possível observar que o uso de matrizes de afinidade possibilita à ELM lidar com a situação comum de escassez de rótulos disponíveis para treinamento.

5

CONCLUSÕES E TRABALHOS FUTUROS

Este capítulo apresenta as conclusões sobre o método desenvolvido e sobre os resultados obtidos, assim como as oportunidades de continuidade de desenvolvimento deste trabalho.

5.1 CONCLUSÕES

Um dos princípios da Máquina de Aprendizado Extremo - *Extreme Learning Machine* (ELM) (Huang et al., 2004, 2006a) é a projeção em altas dimensões, baseando-se no Teorema de Cover (Cover, 1965), de forma a se obter uma maior probabilidade de separabilidade linear no espaço de características. A teoria da ELM mostra que essa projeção pode ser feita sem necessidade de ajustar parâmetros, ou seja, os pesos da camada escondida podem ser atribuídos aleatoriamente, resultando em uma projeção aleatória no espaço de características de alta dimensionalidade. Assim, o treinamento da ELM é feito em duas etapas, possibilitando que se intervenha no mesmo, adicionando, por exemplo, matrizes de afinidade na projeção aleatória dos padrões. Dado que a projeção é feita em um espaço de alta dimensão, a rede é naturalmente sobredimensionada, aumentando a chance de acontecer sobreajuste (*overfitting*). Para solucionar esse problema, podem ser aplicadas técnicas de regularização.

Nesse sentido, esta tese apresentou um método, chamado de ELM Regularizada com Matrizes de Afinidade - *Affinity Matrix Regularized ELM* (AMR-ELM), que possibilita obter um efeito de regularização de Tikhonov (Tikhonov, 1963) nas ELMs, utilizando informação estrutural sobre os dados obtida *a priori* por meio de matrizes de afinidade. Além do efeito de regularização, importante para que a resposta da rede seja suave, lidando assim com o *overfitting* que pode ocorrer em uma rede sobredimensionada como a ELM, o método desenvolvido também permite que as ELMs sejam utilizadas em um contexto de aprendizado semissupervisionado, dado que, além de inserir informação estrutural na projeção dos dados no espaço de características, o método ainda transforma os pesos da camada de saída de seu valor original em uma soma ponderada de todos os demais rótulos, inserindo uma estimativa, baseada na afinidade entre os padrões, para os rótulos ausentes.

O uso das matrizes de afinidade no aprendizado da ELM permitiu, portanto, melhorar a acurácia desta em problemas de aprendizado supervisionado na maioria dos cenários avaliados, prevenindo sobreajuste (*overfitting*) mesmo quando o número de neurônios (ℓ) cresce. Quando comparada com outras duas versões incrementadas da ELM, a OP-ELM

(Miche et al., 2008, 2010a) e a ELM-Reg (Huang et al., 2012), pode-se observar que, apesar de haver equivalência estatística no desempenho de classificação, a AMR-ELM, ao utilizar uma matriz de afinidade não-paramétrica (matriz de similaridade de cossenos), realiza o treinamento em um tempo bastante inferior na maioria dos cenários. Além disso, pode-se notar que mesmo o número ℓ de neurônios na camada escondida não necessita de ser selecionado caso se escolha um valor grande para o mesmo. Esse resultado é importante pois mostra que a abordagem de regularização do método AMR-ELM permite obter capacidade de generalização compatível com os demais métodos sem necessidade de ajustar parâmetros.

Já no cenário do aprendizado semissupervisionado, o qual é bastante comum, tendo em vista a facilidade de se obter grandes quantidades de dados a um custo relativamente baixo, em contraste com o custo relativamente alto de se obter rótulos para todos esses dados, foi possível mostrar que a forma com que o método AMR-ELM insere a informação estrutural fornecida pelas matrizes de afinidade no treinamento da ELM possibilita que sejam alcançados resultados adequados de acurácia mesmo quando a porcentagem de padrões rotulados é pequena em relação aos padrões disponíveis para treinamento - por exemplo, 10% ou 20%. Assim, essa informação estrutural não-supervisionada fez importante diferença para melhorar o desempenho da ELM também em um contexto de escassez de rótulos.

Um resumo dos trabalhos futuros é apresentado na seção seguinte.

5.2 TRABALHOS FUTUROS

Como propostas de continuidade deste trabalho, sugere-se investir nos seguintes assuntos relacionados:

- Baseando-se na ideia da ELM Sequencial Online - *Online Sequential Extreme Learning Machine* (OS-ELM), avaliar a possibilidade de se implementar uma versão *online* do método, a qual possibilite inserir constantemente informações de afinidade ao modelo já treinado, incorporando informações de novos padrões. Em tese, ao longo do tempo de vida da rede, a mesma terá cada vez mais informações estruturais sobre os dados incorporadas, permitindo uma melhor capacidade de predição;

- Estender o desenvolvimento teórico do método para matrizes de afinidade não-normalizadas, de forma a remover uma restrição do método e ampliar as possibilidades de usos de afinidades. Resultados preliminares, apresentados em (Silvestre e Braga, 2014), mostram que matrizes não normalizadas como o produto interno podem ser utilizadas com bons resultados;

- Estudar como as características das bases de dados influenciam na geração das matrizes de afinidade e como, por sua vez, tais

características influenciam no método desenvolvido. Uma característica importante a ser investigada, por exemplo, é a coerência entre agrupamentos e a rotulação;

- Avaliar a possibilidade de estender a ideia de uso de matrizes de afinidade a outros tipos de redes neurais que não as ELMs, assim como nas SVMs, criando assim um *framework* de regularização via matrizes de afinidade para aprendizado de máquina;
- Avaliar o uso de matrizes de afinidade em outros contextos, como, por exemplo, o das classes desbalanceadas;
- Alterar o método para que o mesmo possa lidar com classificação multiclasses.

Parte I

APÊNDICE

APÊNDICE

Este apêndice apresenta os gráficos com os resultados de classificação para as bases reais, para o cenário do aprendizado supervisionado e também para o cenário do aprendizado semissupervisionado. Esses resultados foram apresentados sob forma de tabelas no Capítulo 4.

A.1 APRENDIZADO SUPERVISIONADO

As figuras a seguir apresentam os resultados de classificação para as bases de dados reais utilizadas nos experimentos, conforme descrito no Capítulo 4, para o cenário supervisionado, detalhado na Seção 4.3.1. Para os valores ausentes da OP-ELM, conforme Tabela 4.3, foi calculada a média dos demais valores do mesmo algoritmo.

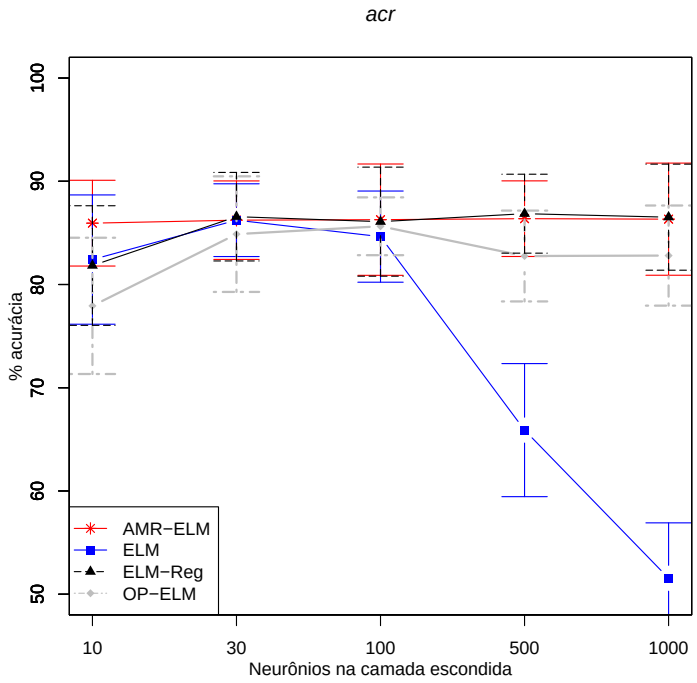


Figura A.1: Acurácia de teste para a base de dados `acr` - cenário supervisionado.

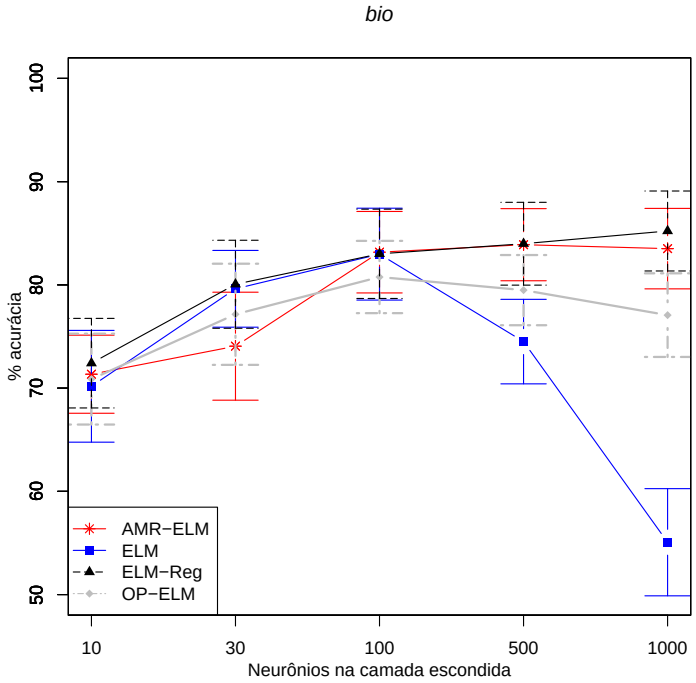


Figura A.2: Acurácia de teste para a base de dados `bio` - cenário supervisionado.

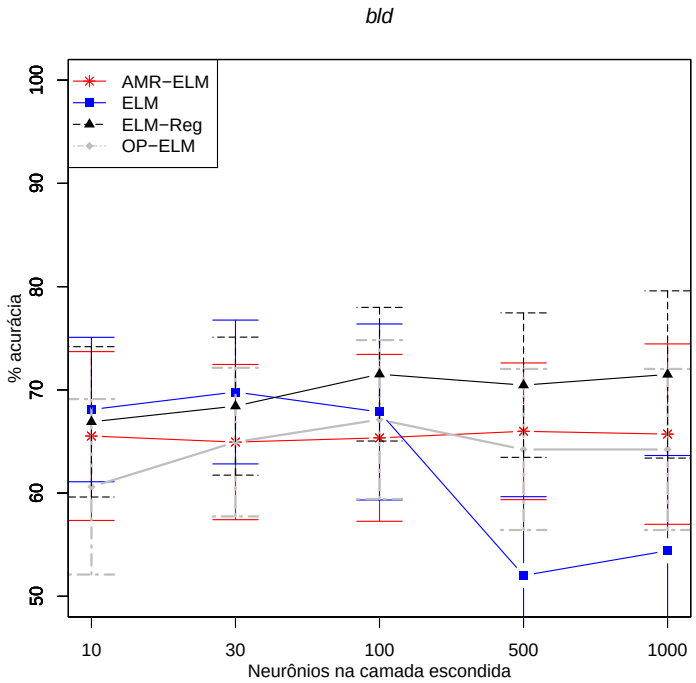


Figura A.3: Acurácia de teste para a base de dados `bld` - cenário supervisionado.

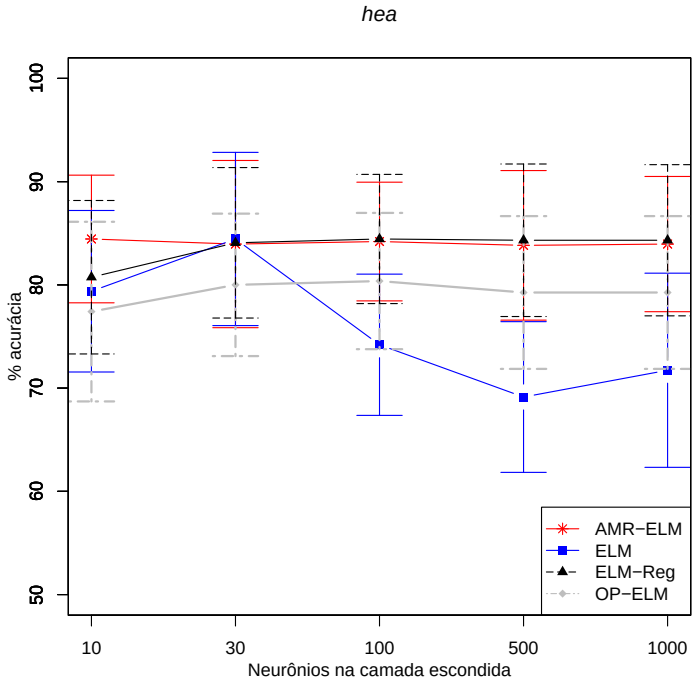


Figura A.4: Acurácia de teste para a base de dados `hea` - cenário supervisionado.

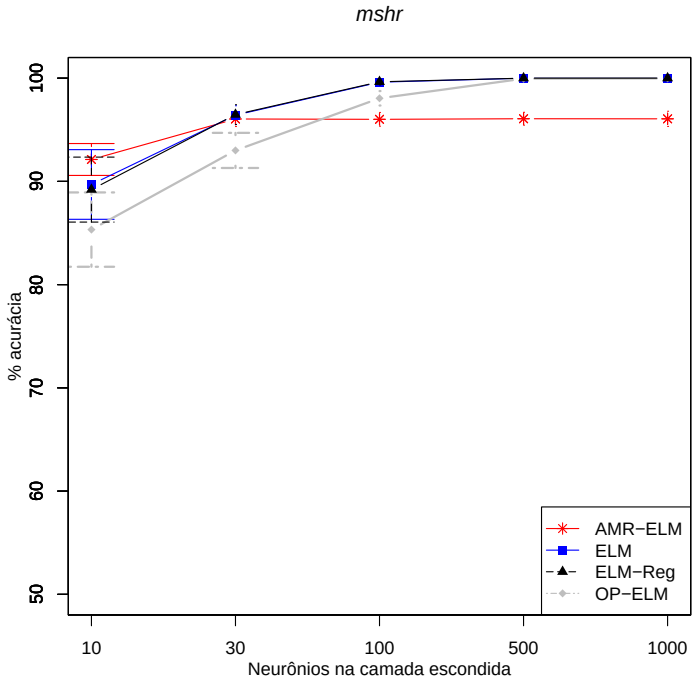


Figura A.5: Acurácia de teste para a base de dados `mshr` - cenário supervisionado.

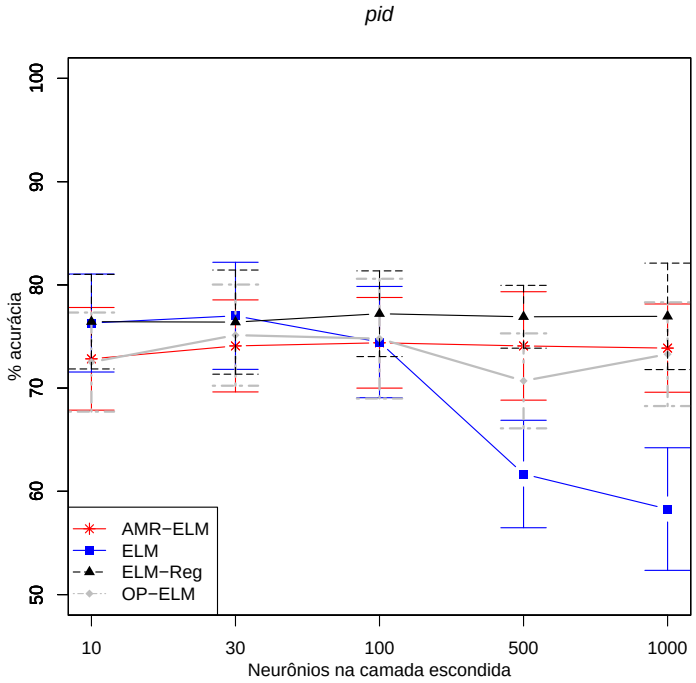


Figura A.6: Acurácia de teste para a base de dados `pid` - cenário supervisionado.

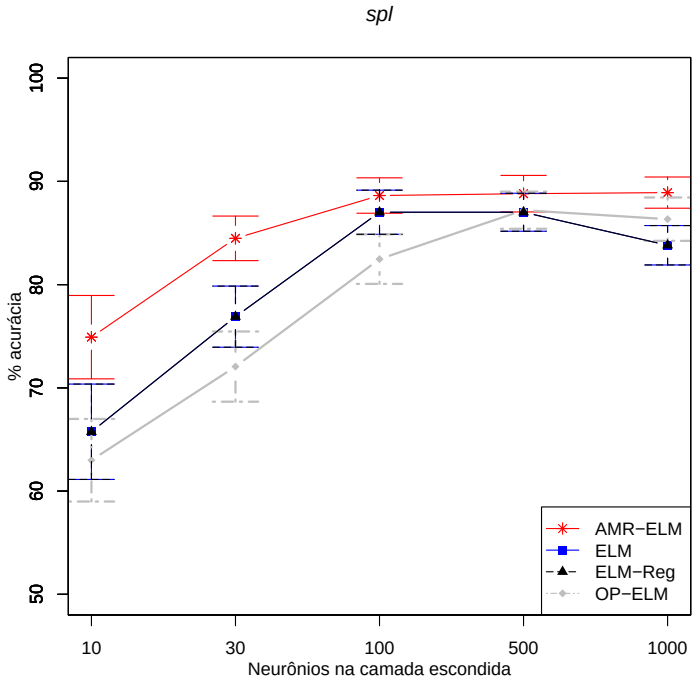


Figura A.7: Acurácia de teste para a base de dados `spl` - cenário supervisionado.

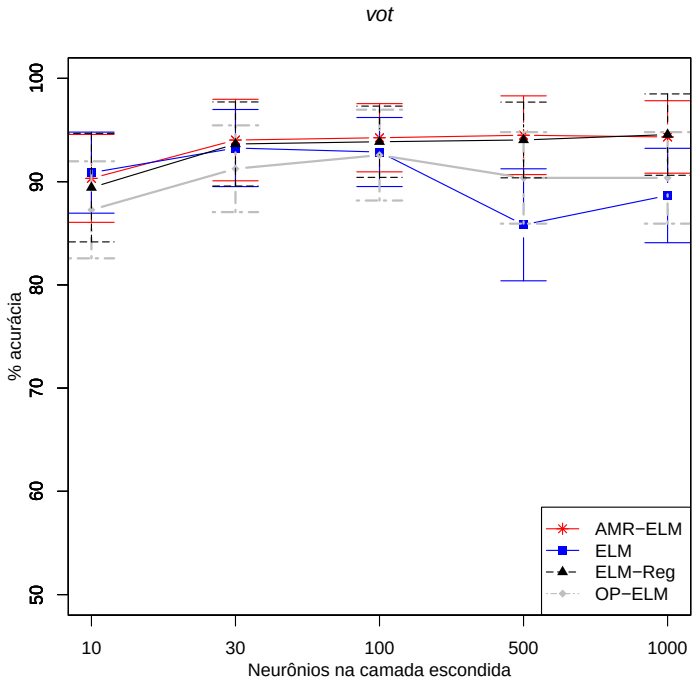


Figura A.8: Acurácia de teste para a base de dados `vot` - cenário supervisionado.

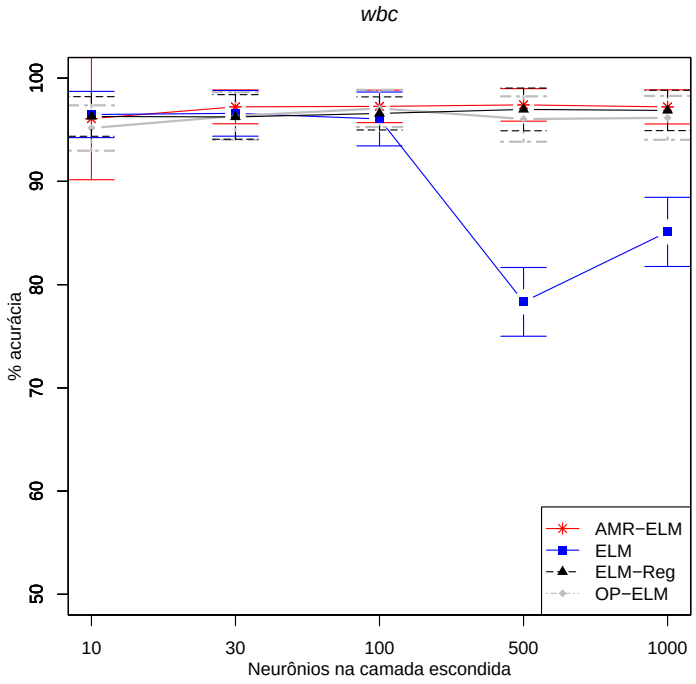


Figura A.9: Acurácia de teste para a base de dados `wbc` - cenário supervisionado.

A.2 APRENDIZADO SEMISSUPERVISIONADO

As figuras a seguir apresentam os resultados de classificação para as bases de dados reais utilizadas nos experimentos, conforme descrito no Capítulo 4, para o cenário semissupervisionado, detalhado na Seção 4.3.2. Para os valores ausentes da OP-ELM, conforme Tabela 4.7, foi calculada a média dos demais valores do mesmo algoritmo.

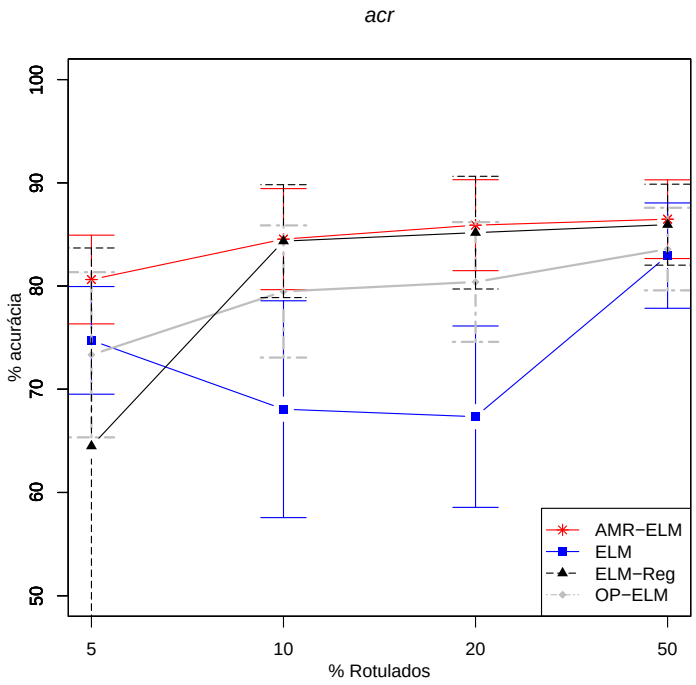


Figura A.10: Acurácia de teste para a base de dados `acr` - cenário semisupervisionado.

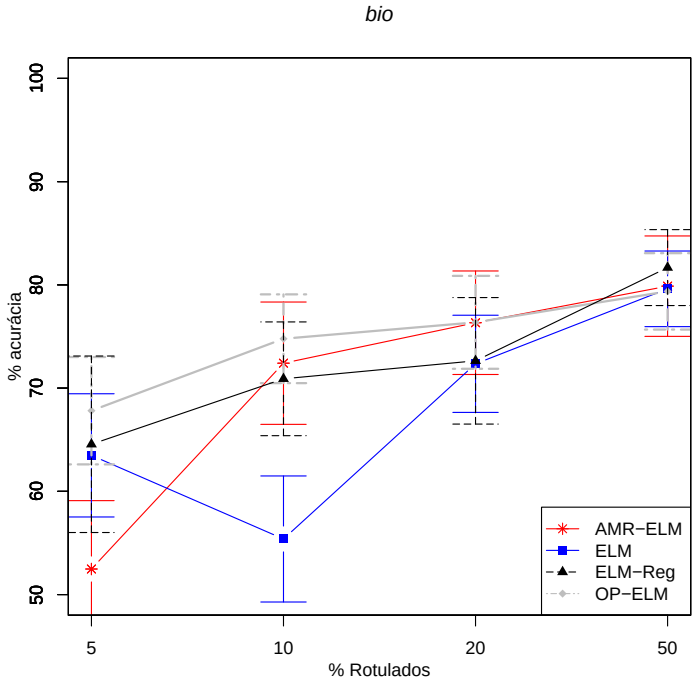


Figura A.11: Acurácia de teste para a base de dados `bio` - cenário semisupervisionado.

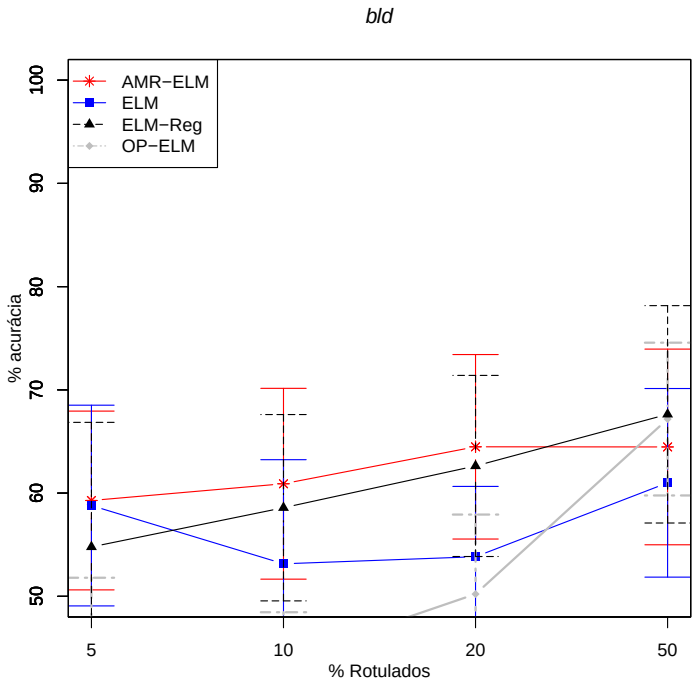


Figura A.12: Acurácia de teste para a base de dados `bld` - cenário semisupervisionado.

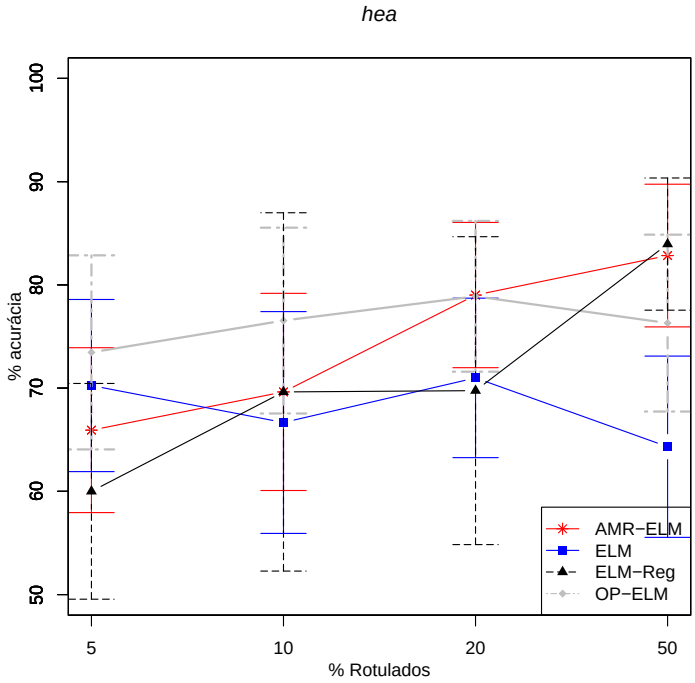


Figura A.13: Acurácia de teste para a base de dados `hea` - cenário semisupervisionado.

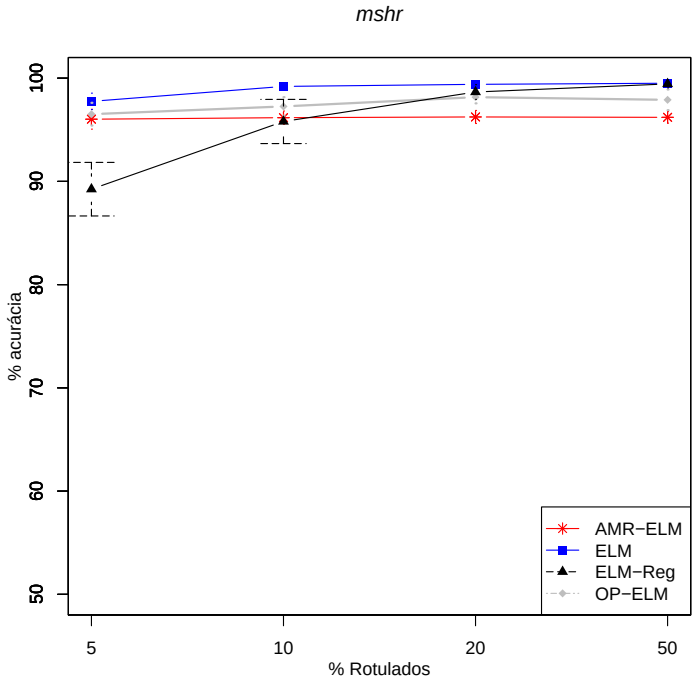


Figura A.14: Acurácia de teste para a base de dados `mshr` - cenário semisupervisionado.

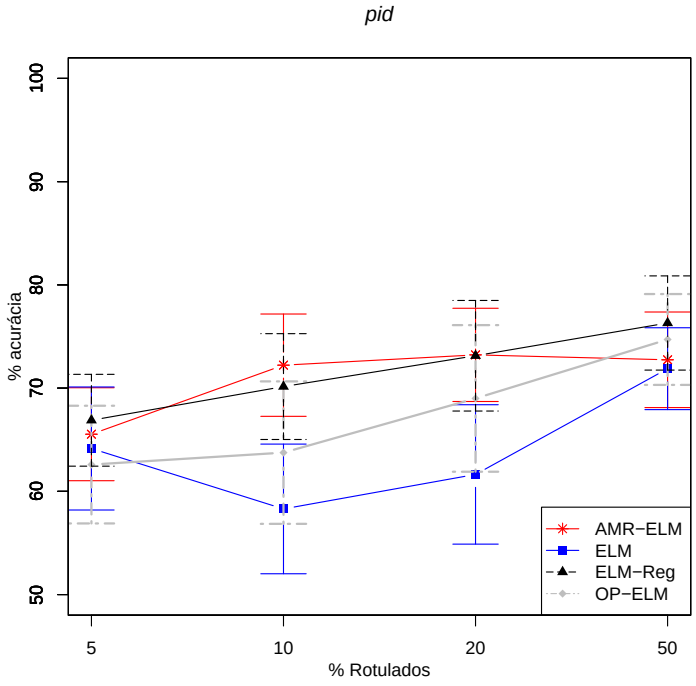


Figura A.15: Acurácia de teste para a base de dados `pid` - cenário semisupervisionado.

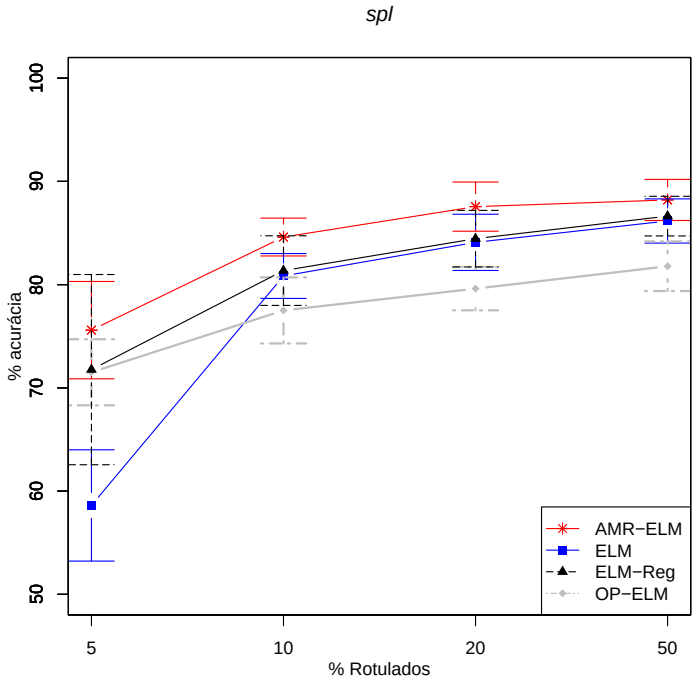


Figura A.16: Acurácia de teste para a base de dados `spl` - cenário semisupervisionado.

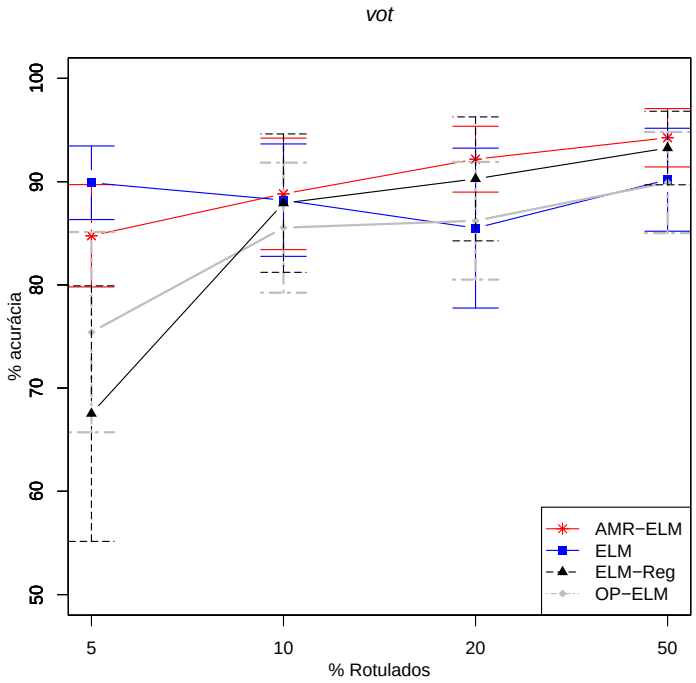


Figura A.17: Acurácia de teste para a base de dados `vot` - cenário semisupervisionado.

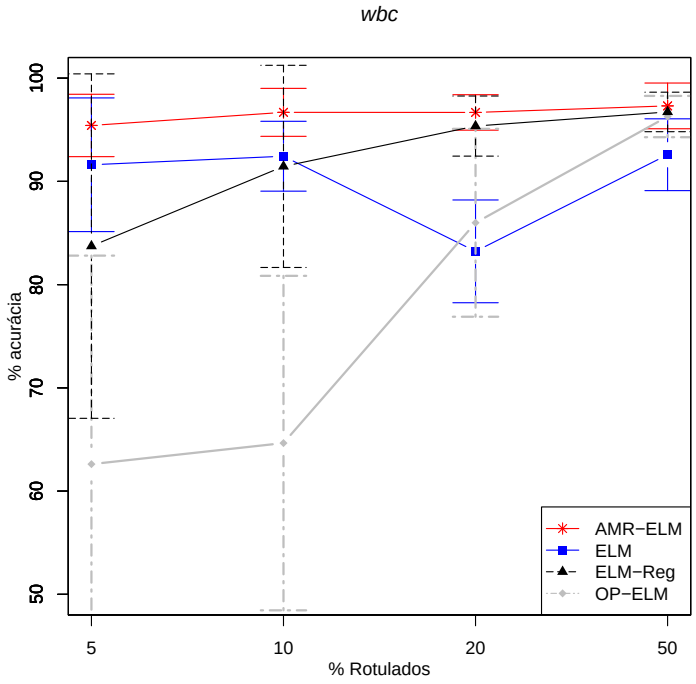


Figura A.18: Acurácia de teste para a base de dados `wbc` - cenário semisupervisionado.

REFERÊNCIAS BIBLIOGRÁFICAS

Asuncion, A. e Newman, D. (2007). UCI machine learning repository.

Bandyopadhyay, S. e Saha, S. (2013). *Unsupervised Classification - Similarity Measures, Classical and Metaheuristic Approaches, and Applications.* Springer.

Barros, A. L. B. d. P. e Barreto, G. A. (2013). Building a robust extreme learning machine for classification in the presence of outliers. Em *Hybrid Artificial Intelligent Systems - 8th International Conference, HAIS 2013, Salamanca, Spain, September 11-13, 2013. Proceedings*, pgs. 588–597.

Bartlett, P. (1998). The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *Information Theory, IEEE Transactions on*, 44(2):525 –536.

Belkin, M., Matveeva, I., e Niyogi, P. (2004). Regularization and semi-supervised learning on large graphs. Em *COLT*, pgs. 624–638.

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

Castro, C. L. d. (2011). *Novos critérios para seleção de modelos neurais em problemas de classificação com dados desbalanceados.* PhD thesis, Universidade Federal de Minas Gerais.

Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307.

Chapelle, O., Schölkopf, B., e Zien, A., editores (2006). *Semi-Supervised Learning.* MIT Press, Cambridge, MA.

Chen, Z. e Haykin, S. (2002). On different facets of regularization theory. *Neural Computation*, 14(12):2791–2846.

Cortes, C. e Vapnik, V. (1995). Support-vector networks. Em *Machine Learning*, pgs. 273–297.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, 14:326–334.

Cristianini, N. e Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

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

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Deng, W., Zheng, Q., e Chen, L. (2009). Regularized extreme learning machine. Em *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on*, pgs. 389–395. IEEE.

Duda, R. O., Hart, P. E., e Stork, D. G. (2000). *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition.

Feng, G., Huang, G.-B., Lin, Q., e Gay, R. K. L. (2009). Error minimized extreme learning machine with growth of hidden nodes and incremental learning. *IEEE Transactions on Neural Networks*, 20(8):1352–1357.

Ferreira, V. H. (2005). Técnicas de regularização de modelos neurais aplicadas à previsão de carga a curto prazo. Master's thesis, Universidade Federal do Rio de Janeiro.

Frénay, B. e Verleysen, M. (2011). Parameter-insensitive kernel in extreme learning for non-linear support vector regression. *Neurocomputing*, 74(16):2526–2531.

Friedman, M. (1937). The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, 32(200):675–701.

Friedman, M. (1940). A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, 11(1):86–92.

Frénay, B. e Verleysen, M. (2010). Using svms with randomised feature spaces: an extreme learning approach. Em *ESANN*.

Girosi, F., Jones, M., e Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269.

Golub, G. H. e Van Loan, C. F. (1996). *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

Goshtasby, A. (2012). *Image Registration: Principles, Tools and Methods*. Advances in Computer Vision and Pattern Recognition. Springer.

Hadamard, J. (1902). Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52.

Hastie, T., Tibshirani, R., e Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. Macmillan, New York.

Hoerl, A. E. e Kennard, R. W. (1970). Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12:69–82.

Huang, G., Zhu, Q., e Siew, C. (2006a). Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1-3):489–501.

Huang, G.-B., Chen, L., e Siew, C. K. (2006b). Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks*, 17(4):879–892.

Huang, G.-B., Liang, N.-Y., Rong, H.-J., Saratchandran, P., e Sundararajan, N. (2005). On-line sequential extreme learning machine. Em *Computational Intelligence*, pgs. 232–237.

Huang, G.-B., Zhou, H., Ding, X., e Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 42(2):513–529.

Huang, G.-B., Zhu, Q.-Y., e Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. Em *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 2, pgs. 985–990 vol.2.

Japkowicz, N. e Shah, M. (2011). *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, New York, NY, USA.

Karpathy, A. (2015). Cs231n: Convolutional neural networks for visual recognition - lecture notes. Acesso em: 06/03/2015.

Lan, Y., Soh, Y. C., e Huang, G.-B. (2009). Ensemble of online sequential extreme learning machine. *Neurocomputing*, 72(13-15):3391–3395.

Li, K., Zhang, J., Xu, H., Luo, S., e Li, H. (2013). A semi-supervised extreme learning machine method based on co-training. *Journal of Computational Information Systems*, 9:207–214.

Liu, J., Chen, Y., Liu, M., e Zhao, Z. (2011). Selm: Semi-supervised elm with application in sparse calibrated location estimation. *Neurocomputing*, 74(16):2566–2572.

Mangasarian, O. L. e Wild, E. W. (2001). Proximal support vector machine classifiers. Em *Proceedings KDD-2001: Knowledge Discovery and Data Mining*, pgs. 77–86.

Mao, W., Tian, M., Cao, X., e Xu, J. (2012). Model selection of extreme learning machine based on multi-objective optimization. *Neural Computing and Applications*, pgs. 1–9.

Martínez-Martínez, J. M., Escandell-Montero, P., Soria-Olivas, E., Martín-Guerrero, J. D., Magdalena-Benedito, R., e Gómez-Sanchis, J. (2011). Regularized extreme learning machine for regression problems. *Neurocomputing*, 74(17):3716–3721.

MATLAB (2009). *version 7.6.0 (R2008a)*. The MathWorks Inc., Natick, Massachusetts.

Miche, Y., Schrauwen, B., e Lendasse, A. (2010a). Machine learning techniques based on random projections. Em Verleysen, M., editor, *ESANN2010: 18th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pgs. 295–302, Bruges, Belgium. d-side Publications.

Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., e Lendasse, A. (2010b). OP-ELM: Optimally-pruned extreme learning machine. *IEEE Transactions on Neural Networks*, 21(1):158–162.

Miche, Y., Sorjamaa, A., e Lendasse, A. (2008). OP-ELM: Theory, experiments and a toolbox. Em Vera Kurková, R. N. e Koutník, J., editores, *LNCS - Artificial Neural Networks - ICANN 2008 - Part I*, volume 5163/2008 de *Lecture Notes in Computer Science*, pgs. 145–154. Springer Berlin / Heidelberg.

Miche, Y., Van Heeswijk, M., Bas, P., Simula, O., e Lendasse, A. (2011). Trop-elm: A double-regularized elm using lars and tikhonov regularization. *Neurocomputing*, 74(16):2413–2421.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. The MIT Press.

Nemenyi, P. (1963). *Distribution-free multiple comparisons*. PhD thesis, New Jersey, USA.

Poggio, T. e Girosi, F. (1990). Networks for Approximation and Learning. *Proceedings of the IEEE*, 78(9).

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rosales, R. e Frey, B. (2002). Learning generative models of similarity matrices. Em *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pgs. 485–492. Morgan Kaufmann Publishers Inc.

Rumelhart, D., Hintont, G., e Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Scholkopf, B. e Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464.

Serre, D. (2002). *Matrices: Theory and Applications*. Springer, 1st edition. edition.

Silvestre, L. J. e Braga, A. P. (2014). Aprendizado semissupervisionado com extreme learning machines e matrizes de afinidade. Em *Proceedings of 1st BRICS Countries Congress (BRICS-CCI) and 11th Brazilian Congress (CBIC) on Computational Intelligence*, CBIC.

Silvestre, L. J., Lemos, A. P., Braga, J. P., e Braga, A. P. (2014). Parameter-free regularization in extreme learning machines with affinity matrices. Em *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, ESANN, pgs. 595–600.

Suykens, J. A. K. e Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Process. Lett.*, 9(3):293–300.

Theodoridis, S. e Koutroumbas, K. (2008). *Pattern Recognition, Fourth Edition*. Academic Press, 4th edition.

Tikhonov, A. N. (1963). Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*

Toh, K. A. (2008). Deterministic neural classification. *Neural Comput*, 20(6):1565–1595.

Vapnik, V. N. (1999). An overview of statistical learning theory. *Trans. Neur. Netw.*, 10(5):988–999.

Velho, H. F. d. C. (2001). Problemas inversos: conceitos básicos e aplicações.

Yin, J., Dong, F., e Wang, N. (2009). Modified gram-schmidt algorithm for extreme learning machine. Em *Proceedings of the 2009 Second International Symposium on Computational Intelligence and Design - Volume 02*, ISCID '09, pgs. 517–520, Washington, DC, USA. IEEE Computer Society.

Yu, Q., Miche, Y., Eirola, E., Van Heeswijk, M., Séverin, E., e Lendasse, A. (2013). Regularized extreme learning machine for regression with missing data. *Neurocomputing*, 102:45–51.

Zhu, Q.-Y., Qin, A. K., Suganthan, P. N., e Huang, G.-B. (2005). Evolutionary extreme learning machine. *Pattern Recognition*, 38(10):1759–1763.

Zhu, X., Goldberg, A. B., Brachman, R., e Dietterich, T. (2009). *Introduction to Semi-Supervised Learning*. Morgan and Claypool Publishers.