Redes Neurais Artificiais: Artigo 2

Victor São Paulo Ruela
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
Email: victorspruela@ufmg.br

***Resumo*— Este trabalho tem como objetivo avaliar o desempneho de diferentes modelos de redes neurais artificiais estudados durante a disciplina sobre bases de dados de benchmark presentes na literatura. Serão considerados o Perceptron, Adaline, Redes RBF, ELM e ELM com aprendizado Hebbiano. Para três problemas de regressão e classificação binária escolhidos, um experimento foi desenhado seguindo as recomendações da literatura. Os resultados de cada modelos são comparados por meio de testes estatíscticos para as metrícas AUC (classficação) e coeficiente de correlação linear (regressão). Os resultados mostraram que, em geral, o ELM obteve um excelente desempenho para todas as bases de dados consideradas em relação aos demais modelos avaliados. Além disso, ficou evidente a dificuldade de se trabalhar com o RBF, o qual obteve os piores resultados para classificação. Bons resultados também foram obtidos para o ELM Hebbiano, mostrando o potencial desta abordagem para problemas de classificação.**

I. Introdução

A Rede Neural Artificial (RNA) é uma classe de modelos muito popular em problemas de classificação, reconhecimento de padrões, regressão e predição [1]. Inspirado pelas características do cérebro humano, elas possuem como elementos básicos neurônios artificiais capazes de executar operações matemáticas, representando desta forma modelos de neurônios biológicos. Através de sua organização em diferentes estruturas de rede, tais modelos são capazes de se adaptar e representar funções matemáticas bastante complexas.

Neste trabalho será feita uma comparação entre diferentes modelos de RNAs: Perceptron, Adaline, Redes RBF, ELM e ELM com aprendizado Hebbiano. Para isso, um experimento será desenhado para comparar estatisticamente o seu desempenho sobre diferentes bases de dados de *benchmark* disponíveis na literatura. Serão considerados tantos problemas de regressão e classificação.

II. Revisão da literatura

Nesta seção é feita uma breve descrição dos modelos de RNAs utilizados neste trabalho.

A. Perceptron

Proposto inicialmente por Rosenblatt [2], este é um modelo geralmente utilizado para a solução de problemas de classificação lineares. No seu trabalho original, o autor descreve formas de adaptação dos parâmetros, ou pesos, da rede com o objetivo de reduzir a discrepância entre as saídas esperadas e estimadas e aprender associações entre os neurônios, o que é a base da indução para diversos algoritmos atuais. Este trabalho é considerado um marco na literatura por diversos autores.

Se considerarmos uma função de ativação contínua e diferenciável, os pesos da rede poderão ser inferidos de forma explícita, através do cálculo da pseudo-inversa, ou pelo algoritmo do gradiente descendente [3]. Exemplos de funções de ativação com esta característica frequentemente empregadas na literatura são a função logística, tangente hiperbólica e linear [1]. Vale a pena ressaltar que a covergência destas abordagens está condicionada aos dados utilizados para treinamento serem linearmente independentes [3].

B. Adaline

O Adaline foi inicialmente desenvolvido por Widrow em 1960 [4], sendo principamente aplicado em problemas de regressão lineares. Assim como o Perceptron, originalmente este modelo considera somente um neurônio MCP em sua formulação, entretando sua função de ativação é a identidade. Seu treinamento é formulado como um problema de otimização com custo quadrático, onde originalmente foi utilizado o algoritmo do gradiente descendente para sua solução.

Para este agoritmo, em cada iteração é dado um passo na direção oposta ao gradiente da função objetivo, resultanto em uma convergência gradual para o mínimo do problema. Este gradiente pode ser calculado de forma análitica para a estrutura de rede do Adaline, o qual é no fim proporcional à diferença entre os valores estimados e reais [4], bastante similar ao Perceptron simples. É fácil notar que o treinamento também pode ser realizado de forma direta através do cálculo da pseudo-inversa dos dados de entrada, já que este é um problema de mínimos quadrados [5].

C. Redes RBF

As redes RBF foram inicialmente introduzidas por [6] e são caracterizadas por um aprendizado que envolve duas etapas: (i) aplicar uma transformação aos padrões para um espaço onde a probabilidade de serem linearmente separáveis é alta (ii) encontrar os pesos usando o estimador mínimos quadrados usado no Perceptron simples. Essa estrutura pode ser representada por um rede de três camadas, onde sua camada escondida é reponsável pela transformação não-linear das estradas para o novo espaço, geralmente para uma dimensão muito alta.

Essa transformação é justificada pelo teorema de Cover sobre a separabilidade de padrões [7], o qual diz que um problema de classificação complexo projetado não-linearmente

para um espaço de alta dimensão é mais provável de ser separável do que em um espaço de baixa dimensão, desde que o espaço não seja densamente povoado. Boa parte da teoria, que é relacionada ao campo de interpolação multivariável, considera um kernel baseado na função Gaussiana, que é uma classe importante de RBFs. Teoricamente, as redes RBF podem ser consideradas um aproximador universal de funções contínuas se a RBF é selecionada apropriadamente [8], [9], [10].

D. ELM

Inicialmente proposto por [11], as máquinas de aprendizado extremo (ELM) são redes neurais com uma única camada escondida, as quais possuem o atrativo de poucos parâmetros a serem ajustados, generalização maior ou similar e redução do tempo de treinamento das redes em relação aos métodos convencionais. Seu treinamento é baseado na teoria de minimização de risco empírico, necessitando de somente uma iteração para este processo, evitando múltiplas iterações e algoritmos de otimização local [12].

ELMs são capazes de definir adaptivamente o número neurônios da rede e aleatoriamente escolher os pesos das entradas e viéses da camada escondida [13]. Isso faz com o que a rede possa ser considerada como um sistema linear, o qual pode ser calculado de forma analítica através de uma operação de inversão da matriz de pesos da camada de saídas [13]. Essa característica permite uma drástica redução do esforço computacional do treinamento, geralmente de 10 vezes ou mais [14].

III. Metodologia

A. Bases de Dados

Neste trabalho serão consideradas três bases de dados referentes a problemas de classificação binários e regressão multivariada disponíveis no repositório da UCI [15], totalizando seis problemas. Antes do treinamento, os dados de entrada serão normalizados para o intervalo $[-1, 1]$ e filtrados para remoção de valores inválidos. Um sumário das bases de dados consideradas pode ser vista na Tabela I.

TABELA I
PRINCIPAIS CARACTERÍSTICAS DAS BASES DE DADOS UTILIZADAS

	Instâncias	**Atributos**	**Proporção**
Breast Cancer	569	32	0.66
Liver Disorder	245	6	0.58
Statlog (Heart)	270	13	0.66
Boston Housing	506	13	N/A
Wine Quality (Red)	1599	11	N/A
Diabetes	442	10	N/A

B. Desenho do experimento

A partir das recomendações para desenho de experimento para comparação de algoritmos proposta em [16], a seguinte metodologia será adotada:

- Para cada base de dados:
  1) Particionar os dados D em k partições para validação cruzada, mantendo a mesma proporção entre os rótulos
     a) Criar o conjunto de treino $T = D - k$
     b) Para cada modelo:
        i) Executar busca exaustiva com validação cruzada sobre T para os coeficientes de regularização λ
        ii) Escolher λ que obtém o melhor ajuste médio
        iii) Avaliar a métrica do modelo sobre k
  2) Estimar o intervalo de confiança de 95% do valor médio da métrica sobre k usando *bootstraping*

O número de partições consideradas será de $k = 10$. Para o item 1(b)i, serão considerados um número fixo de valores igualmente espaçados dentro de um intervalo pré-definido. Além disso, serão considerados cinco partições para a validação cruzada, como forma de controlar um pouco o tamanho do experimento. Serão consideradas as métricas AUC para classificação e erro quadrático médio (MSE) para regressão, respectivamente.

A implementação deste experimento, bem como dos modelos utilizados, será feita em Python e utilizando principalmente os pacotes *numpy* [17] e *scikit-learn* [18]. Os algoritmos ELM e RBF foram implementados seguindo as notas de aula do professor, sendo que o RBF irá considerar o *k-means* para o cálculo dos centros e raios. O Percepton utilizado está disponível na biblioteca *scikit-learn* diretamente, suportando o uso de regularização. Já o Adaline foi implementado usando o algoritmo MLP disponível no *scikit-learn*, considerando um único neurônio na camada escondida e aprendizado via o algoritmo do gradiente estocástico. Isso foi necessário uma vez que esta implementação suporta o uso de regularização. O experimento será realizado em um Notebook Intel Core i7 Quad Core com 8Gb de memória RAM, sendo que o uso de paralelização será utilizado sempre que possível visto a enorme quantidade de vezes que os modelos serão treinados.

C. ELM Hebbiano

Uma variação presente na literatura para controlar a generalização do ELM consiste no uso do aprendizado Hebbiano após a camada escondida [19]. Conforme sugerido pelo autor, podemos substituir o cálculo da pseudo-inversa da matriz de projeção aleatória por um Perceptron Hebbiano com pesos normalizados. Será considerada o algoritmo de aprendizado Hebbiano considerando somente um neurônio, similiar ao Perceptron simples. Essa abordagem é melhor descrita em [20], da qual podemos retirar a seguinte regra para problemas de classificação binários:

$$w = \frac{\sum_{i=1}^{N} y_i \mathbf{h}_i}{\left\| \sum_{i=1}^{N} y_i \mathbf{h}_i \right\|} \tag{1}$$

onde $\mathbf{h}_i$ é á i-ésima linha da matriz de projeção aleatória do algoritmo ELM original. É importante ressaltar que esta regra assume que os dados foram normalizados para possuir média zero e desvio padrão unitário. Para validar a implementação, o

algoritmo foi executado sobre duas bases de dados simples e 100 neurônios na camada escondida, cujos resultados podem ser vistos na Figura 1.

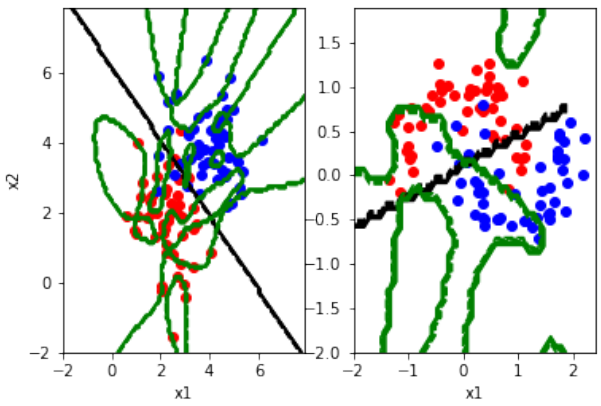

Fig. 1. Comparação entre o ELM original (verde) e sua versão com aprendizado Hebbiano (preto) em um problema linearmente e outro não-linearmente separável

Através destes resultados, podemos concluir que o uso do aprendizado Hebbiano é capaz de atingir a regularização desejada. Entretanto, nota-se que a superfície de separação torna-se predominantemente linear, de forma que seja esperado que esta abordagem possua desempenho limitado para problemas que não sejam linearmente separáveis.

IV. Resultados

A. Problemas de Classificação

Para os problemas de classificação foram considerados os algoritmos Perceptron, RBF, ELM e ELM com apredizado Hebbiano. Foi estabelecido um número fixo de 20 neurônios na camada escondida e não foi aplicada regularização ao ELM Hebbiano. 50 valores para o coeficiente de regularização foram escolhidos no intervalo $[0, 1]$. Os gráficos boxplot para cada conjunto de dados considerado é exibido nas Figuras 2, 3 e 4. Os intervalos de confiança calculados são exibidos na Tabela II.

A partir destes resultados, podemos chegar às seguintes conclusões:

- Para um intervalo de confiança de 95%, podemos afirmar que todos os algoritmos possuem desempenho médio superior ao modelo RBF nas bases de dados avaliadas.
- Para a base de dados *Breast Cancer*, os modelos ELM e Perceptron obtiveram desempenhos bastante similares. Além disso, podemos afirmar que eles obtiveram desempenho médio superior ao ELM Hebbiano para o intervalo de confiança de 95%.
- Para a base de dados *Statlog (Heart)* não podemos rejeitar a hipótese de que os modelos ELM, ELM Hebbiano e Perceptron possuam desempenhos iguais para o intervalo de confiança de 95%.

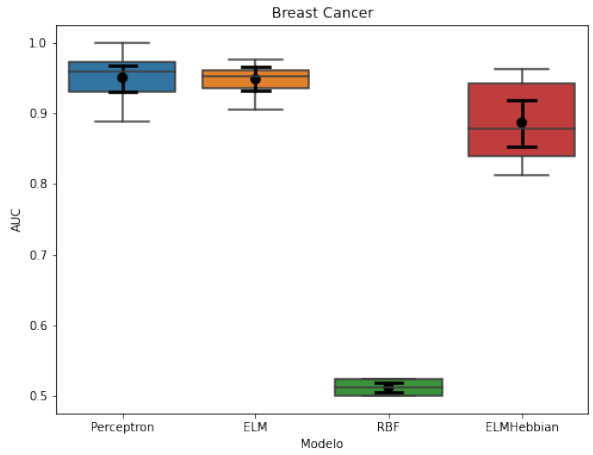

Fig. 2. Boxplots para a base de dados *Breast Cancer*. Intervalos de confiança de 95% para a média são exibidos em preto.

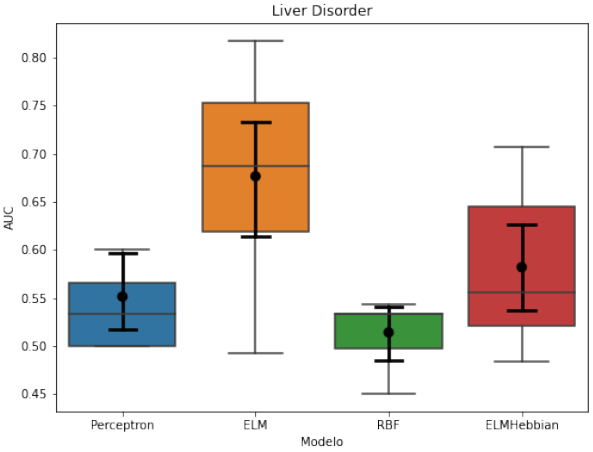

Fig. 3. Boxplots para a base de dados *Liver Disorder*. Intervalos de confiança de 95% para a média são exibidos em preto.

- Para a base de dados *Liver Disorder*, podemos afirmar que para um intervalo de confiança de 95%, o ELM possui desempenho superior aos demais modelos.
- Uma hipótese para o desempenho muito abaixo do esperado do RBF pode estar no número de neurônions escolhidos para a camada escondida. Este hiperparâmetro não foi ajustado para que a comparação com o ELM seja justa e também para limitar o tamanho do experimento a ser executado.
- A diferença de desempenho entre o ELM e sua versão com aprendizado Hebbiano está no fato deste última poder estár resultando em uma regularização excessiva. Isso sugere que o uso de um modelo Hebbiano diferente é indicado para controlar melhor a sua generalização.

TABELA II

INTERVALOS DE CONFIANÇA DE 95% CALCULADOS PARA A AUC MÉDIA

	ELM	ELM Hebbiano	Perceptron	RBF
Breast Cancer	**0.948 (0.935,0.963)**	0.887 (0.853,0.920)	**0.950 (0.930,0.971)**	0.512 (0.505,0.519)
Liver Disorder	**0.677 (0.621,0.733)**	0.582 (0.536,0.628)	0.532 (0.509,0.553)	0.515 (0.498,0.537)
Statlog (Heart)	**0.812 (0.769,0.851)**	0.762 (0.698,0.819)	0.772 (0.700,0.846)	0.547 (0.522,0.569)

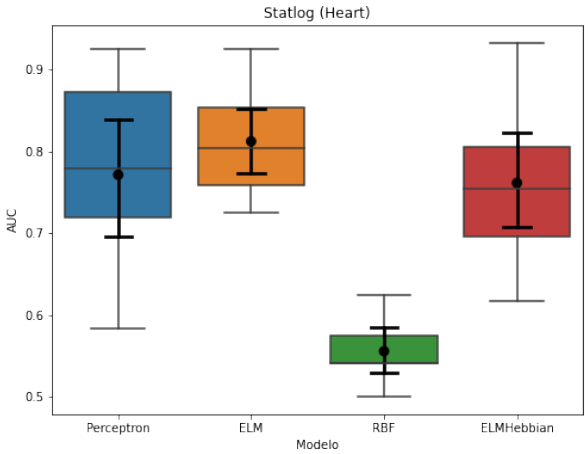

Fig. 4. Boxplots para a base de dados *Statlog (Heart)*. Intervalos de confiança de 95% para a média são exibidos em preto.

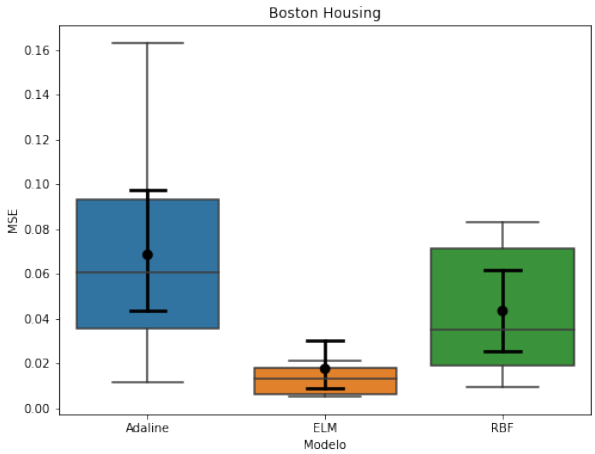

Fig. 5. Boxplots para a base de dados *Boston Housing*. Intervalos de confiança de 95% para a média são exibidos em preto.

B. Problemas de Regressão

Para os problemas de classificação foram considerados os algoritmos Adaline, RBF e ELM. Não foi possível realizar uma implementação do ELM Hebbiano que funcionasse com problemas de regressão, logo este modelo teve que ser descartado. Foi estabelecido um número fixo de 20 neurônions na camada escondida e a regularização foi considerada somente para todos os algoritmos. 50 valores de coeficiente de regularização foram escolhidos no intervalo $[0, 1]$. Nenhum outro ajuste de hiper-parâmetro foi feito para o Adaline, sendo usados os valores padrão da biblioteca *scikit-learn*. Os gráficos boxplot para cada conjunto de dados considerado é exibido nas Figuras 5, 6 e 7. Os intervalos de confiança calculados são exibidos na Tabela II.

A partir destes resultados, podemos chegar às seguintes conclusões:

- Para um intervalo de confiança de 95%, podemos afirmar que o ELM teve desempenho médio superior aos demais algoritmos em todas as bases de dados avaliadas.
- Não podemos rejeitar a hipótese de que os algoritmos RBF e Adaline possuam desempenhos médios iguais para todas as bases de dados, considerando o intervalo de confiança de 95%

V. CONCLUSÕES

Neste trabalho foi feita uma avaliação do desempenho dos modelos ELM, RBF, Adaline, Perceptron e ELM Hebbiano sobre bases de dados de benchmark presentes na literatura. A partir de um experimento desenhado para tal finalidade, os resultados foram comparados estatísticamente utilizando a técnica de *bootstrapping* para a média das métricas AUC e erro quadrático médio, considerando um intervalo de confiança de 95%. Destaca-se o excelente desempenho do ELM em todos os conjuntos de dados, tanto de regressão e classificação. É importante destacar também o desempenho muito abaixo do esperado para o RBF nos problemas de classificação, sugerindo a necessidade de um melhor ajuste do número de neurônios na camada escondida, por exemplo. Notou-se também que os modelos lineares avaliados obtiveram bons resultados dependendo da base de dados em questão, conseguindo atingir um desempenho estatisticamente equivalente aos demais.

Uma surpresa foi o ELM Hebbiano, onde embora tenha sido usada uma abordagem bem simples de aprendizado Hebbiano, não apresentou um resultado muito inferior em relação ao ELM original para classificação. Destaca-se o fato de que não foi possível utilizar a estratégia de aprendizado Hebbiano com sucesso em problemas de regressão. Isso sugere que uma abordagem diferente deve ser utilizada nesta situação. Durante a execução do experimento, pôde ser observado também que o modelo RBF possui um tempo de treinamento muito maior que os demais. Isso fez com que ele fosse o gargalo de todo o experimento neste quesito, o que limitou um pouco o potencial de executar um número maior de partições para a validação cruzada, bem como avaliar um intervalo maior para o fator de regularização.

TABELA III

INTERVALOS DE CONFIANÇA DE 95% CALCULADOS PARA O MSE MÉDIO

	Adaline	ELM	RBF
Boston Housing	0.069 (0.039,0.096)	**0.012 (0.008,0.016)**	0.044 (0.025,0.061)
Diabetes	0.078 (0.066,0.091)	**0.029 (0.025,0.033)**	0.054 (0.048,0.061)
Wine Quality (Red)	0.037 (0.031,0.043)	**0.017 (0.016,0.019)**	0.024 (0.022,0.026)

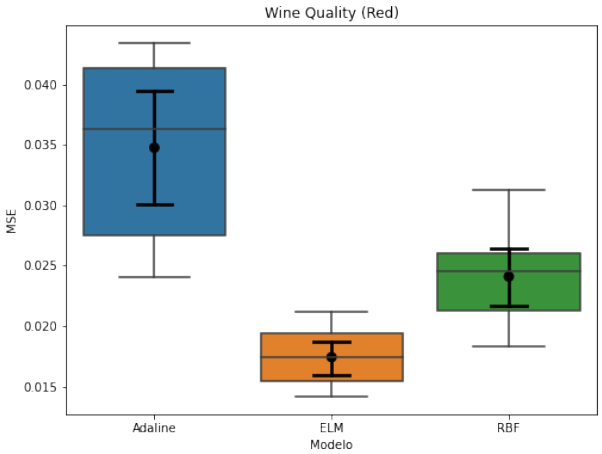

Fig. 6. Boxplots para a base de dados *Wine Quality (Red)*. Intervalos de confiança de 95% para a média são exibidos em preto.

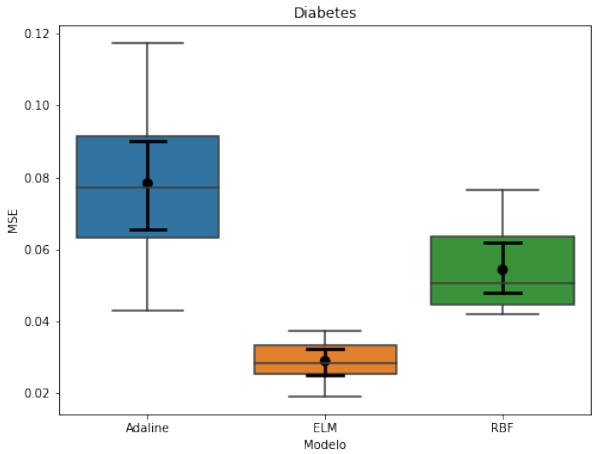

Fig. 7. Boxplots para a base de dados *Diabetes*. Intervalos de confiança de 95% para a média são exibidos em preto.

Como trabalhos futuros, é sugerido um ajuste fino de demais hiper-parâmetros do RBF como forma de tentar melhorar o seu desempenho. Além disso, também é interessante avaliar outras formas de aprendizado Hebbiano que poderiam ser utilizados com o ELM, bem como para melhorar seu poder de generalização e também ser aplicável a problemas de regressão com sucesso. Outra tarefa importante seria a realização de uma etapa de pré-processamento mais completa sobre bases de dados considerados, o qual não foi feita por restrições de tempo mas teria potencial para melhorar os resultados obtidos de todos os modelos.

REFERÊNCIAS

[1] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[2] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[3] John Hertz, Anders Krogh, Richard G Palmer, and Heinz Horner. Introduction to the theory of neural computation. *PhT*, 44(12):70, 1991.

[4] Bernard Widrow and Marcian E Hoff. Adaptive switching circuits. Technical report, Stanford Univ Ca Stanford Electronics Labs, 1960.

[5] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice-Hall, Inc., 2007.

[6] D. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Syst.*, 2, 1988.

[7] Thomas M Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3):326–334, 1965.

[8] Tomaso Poggio and Federico Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[9] Jooyoung Park and Irwin W Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

[10] Yi Liao, Shu-Cherng Fang, and Henry LW Nuttle. Relaxed conditions for radial-basis function networks to be universal approximators. *Neural Networks*, 16(7):1019–1028, 2003.

[11] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, volume 2, pages 985–990. IEEE, 2004.

[12] Shifei Ding, Han Zhao, Yanan Zhang, Xinzheng Xu, and Ru Nie. Extreme learning machine: algorithm, theory and applications. *Artificial Intelligence Review*, 44(1):103–115, 2015.

[13] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[14] Wan-Yu Deng, Qing-Hua Zheng, Lin Chen, and Xue-Bin Xu. Research on extreme learning of neural networks. *Chinese Journal of Computers*, 33(2):279–287, 2010.

[15] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[16] Steven L Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data mining and knowledge discovery*, 1(3):317–328, 1997.

[17] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] Euler Guimarães Horta. Aplicação de máquinas de aprendizado extremo ao problema de aprendizado ativo. 2015.

[20] Manuel Fernandez-Delgado, Jorge Ribeiro, Eva Cernadas, and Senén Barro Ameneiro. Direct parallel perceptrons (dpps): fast analytical

calculation of the parallel perceptrons weights with margin control for classification tasks. *IEEE transactions on neural networks*, 22(11):1837–1848, 2011.