Algoritmo de Treinamento para uma Rede SLFN com Projeção Aleatória e Margem Larga
Learning Algorithm for an SLFN Network with Random Projection and Large Margin

Vítor Gabriel Reis Caitité
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
vcaitite@ufmg.br

Raul Fonseca Neto
Universidade Federal de Juiz de Fora
Juiz de Fora, Brazil
raulfonseca.neto@ufjf.br

Frederico Coelho
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
fredgfc@ufmg.br

Antônio Pádua Braga
Universidade Federal de Minas Gerais
Belo Horizonte, Brasil
apbraga@ufmg.br

ABSTRACT

This work presents a large margin learning algorithm for single hidden layer feedforward networks (SLFNs) with random weights for the hidden neurons, called RP-IMA. This algorithm, applied to binary classification problems, proposes randomly assigned weights to the hidden layer and the use of an incremental margin algorithm to calculate the weights of the output neuron of the SLFN. The results showed that the proposed algorithm is capable to obtain a large separation margin in the feature space and has its performance less sensitive to variations in the network architecture, when compared to Extreme Learning Machines.

KEYWORDS

neural networks, large margin, binary classification, random projection

PALAVRAS-CHAVE

redes neurais, margem larga, classificação binária, projeção aleatória

1 INTRODUÇÃO

Desde a década de 1990 já existiam redes neurais de camada oculta única que possuíam os pesos da camada escondida definidos aleatoriamente [13, 14, 16]. Esse tipo de treinamento, que define uma projeção aleatória na camada oculta, se tornou popular com as Máquinas de Aprendizado Extremo (*Extreme Learning Machine* - ELM) [9, 10].

As ELMs podem ser formalmente definidas como uma solução onde os pesos da camada oculta são selecionados aleatoriamente e os pesos da camada de saída linear são obtidos diretamente usando o método da pseudo-inversa. Essa abordagem tem a vantagem de produzir boa performance de generalização e ainda ser simples e ter custo computacional menor se comparado com outros métodos de treinamento como o *backpropagation*.

Apesar dessas vantagens apresentadas, conforme discutido em [12] a ELM tende a ter mais neurônios ocultos do que os algoritmos convencionais de aprendizado iterativo, o que pode levar ao *overfitting*. É pensando nisso que este trabalho propõe outra técnica de treinamento para redes *feedforward* de camada oculta única (*Single Hidden Layer Feedforward Neural Network* - SLFNs) com projeção aleatória, considerando problemas de classificação binária.

Neste novo método, chamado *Random Projection Incremental Margin Algorithm* (RP-IMA), usamos uma projeção aleatória (na camada oculta de uma SLFN) juntamente com um algoritmo de margem incremental para o cálculo dos pesos do neurônio de saída da SLFN. O neurônio de saída da rede obtém um resultado de margem larga para a camada de saída encontrando sucessivamente a solução para um problema de classificação considerando valores de margem crescentes. A ideia é usar a projeção aleatória para melhorar a separabilidade do problema e então aplicar um algoritmo de margem incremental para obter um classificador de margem larga. Ao maximizar a margem de decisão no espaço de características, espera-se obter um classificador com melhor capacidade de generalização e mais robusto, tendo seu desempenho menos afetado por variações na arquitetura da rede (o que é um problema na implementação original da ELM).

2 TRABALHOS RELACIONADOS

Máquinas de Vetores de Suporte (*Support Vector Machine* - SVM) [2] é possivelmente o método mais conhecido para problemas de classificação binária de margem larga. Seu sucesso deve-se provavelmente à formulação elegante do problema de programação quadrática e ao fato de adotar um mapeamento implícito por *kernel* para o espaço de características.

Com o intuito de evitar o custo computacional associado a problemas de programação quadrática, foram propostos na literatura algoritmos mais simples e eficientes que buscam apenas aproximar a solução de margem máxima [8, 11]. O trabalho [11] apresentou o algoritmo de margem incremental (*Incremental Margin Algorithm* - IMA), que é capaz de obter uma solução de margem larga executando seguidamente o Perceptron de Margem Fixa (*Fixed Margin Perceptron* - FMP). O FMP é uma extensão do Perceptron de Rosenblatt, que visa encontrar a solução de um problema de aprendizagem linearmente separável dada uma margem fixa. Assim, a cada iteração do IMA, incrementa-se a margem fixa fornecida ao FMP e busca-se uma nova solução.

Uma outra contribuição para o campo, apresentada em [22], é baseada no algoritmo k-vizinhos mais próximos (*k Nearest Neighbor* - kNN) [3]. A proposta dos autores foi aprender uma métrica de distância de Mahalanobis para o kNN com o objetivo de que os k vizinhos mais próximos sempre estejam na mesma classe enquanto amostras de classes diferentes são separadas por uma margem larga [22]. Ainda baseado no kNN, o trabalho [4] propôs aplicar a fronteira de decisão de SVMs para induzir uma métrica de distância localmente adaptativa para o kNN.

Outra abordagem importante usada para construir classificadores de margem larga é estudar o problema a partir de uma formulação geométrica. O artigo [18] apresentou um classificador baseado apenas na estrutura dos dados representados pelo Grafo de Gabriel [7], para minimizar o erro do conjunto de treinamento e maximizar a margem de separação entre as classes. Neste método, a superfície de separação resulta de um modelo de misturas de hiperplanos cujos parâmetros são retirados do próprio grafo, mais especificamente dos padrões localizados próximos à margem de decisão. Ainda baseado no Grafo de Gabriel, há uma variedade de outros métodos que implementam classificadores de larga margem. Uma abordagem possível é, por exemplo, a aplicação de um kNN com k=1, porém considerando como conjunto de treinamento apenas os vetores geométricos (obtidos do grafo associado a informações dos rótulo das amostras[19]) [1].

Por fim, o tema de margem larga também tem sido estudado no contexto do aprendizado profundo. Nesse cenário, uma abordagem possível é focar apenas na margem na camada de saída. Por exemplo, o trabalho [23] propôs o uso de uma SVM na camada de saída da rede. Em outra perspectiva, [6] relatou como as abordagens clássicas de maximização de margem não são apropriadas para modelos profundos, sendo que os métodos convencionais apenas impõem a maximização de margem na camada de saída. Os autores também propuseram uma função de perda na qual uma margem larga pode ser imposta em qualquer conjunto de camadas [6].

3 REFERENCIAL TEÓRICO

Nas subseções a seguir serão explicados com mais detalhes alguns algoritmos da literatura que foram fundamentais para o desenvolvimento deste trabalho.

3.1 Máquinas de Aprendizado Extremo

Como descrito em [10], a ELM se constitui como um método de treinamento simples pra redes neurais artificiais de duas camadas, composto basicamente pelas seguintes três etapas:

(1) Gerar aleatoriamente os pesos de entrada $\mathbf{w}_i$ e o termo *bias* b_i, i = 1, ..., Ñ, sendo Ñ o número de neurônios da camada escondida.
(2) Calcular a matriz de mapeamento **H** (saída da camada escondida), tal que:

$$\mathbf{H} = \begin{bmatrix} h(\mathbf{x}_1) \\ \vdots \\ h(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b1) & \dots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b1) & \dots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}_{Nx\tilde{N}}, \quad (1)$$

dado que N é a quantidade de amostras de treinamento, $\mathbf{x}_i$ (com $i = 1, ..., N$) são as amostras e g(x) é a função de ativação utilizada.
(3) Calcular os pesos de saída $\boldsymbol{\beta}$, tal que:

$$\boldsymbol{\beta} = \mathbf{H}^{+}\mathbf{Y}, \quad (2)$$

sendo $\mathbf{H}^{+}$ a pseudoinversa de Moore-Penrose de **H**, e $\mathbf{Y} = [y_1, \dots, y_N]^T$ é o vetor contendo as saídas correspondentes as N amostras de treinamento.

No caso de um problema de classificação binária, a saída de uma ELM, com Ñ neurônios na camada escondida, para uma entrada $\mathbf{x}_i$ pode ser dada por:

$$\hat{y}_i = sign(\sum_{i=1}^{\tilde{N}} \beta_i h_i(\mathbf{x}_i)), \quad (3)$$

sendo a função *sign* definida como mostrado na equação 4.

$$sign(x) = \begin{cases} -1 & \text{se } x < 0 \\ +1 & \text{caso contrário,} \end{cases} \quad (4)$$

A Figura 1 apresenta a topologia típica da ELM quando aplicada a problemas de classificação binária.

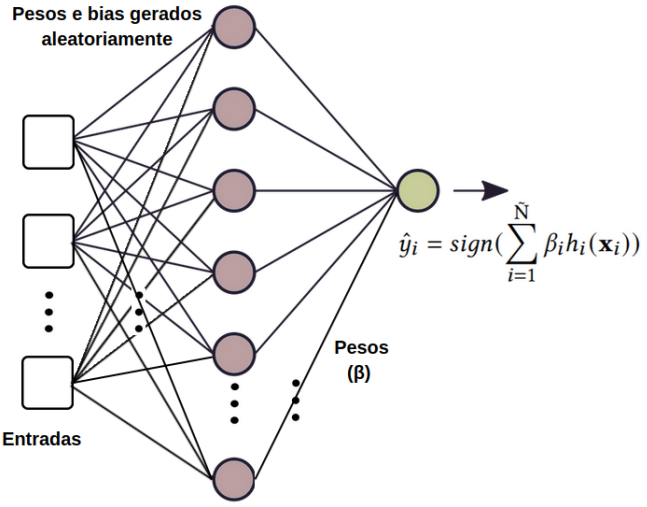

Figura 1: Topologia típica de uma ELM.

3.2 Perceptron de Margem Fixa

O perceptron de margem fixa (*Fixed Margin Perceptron* - FMP) se trata de uma extensão do Perceptron de Rosenblatt [15], que visa encontrar a solução de um problema de aprendizagem linearmente separável dada uma margem fixa.

De acordo com a formulação do FMP [11], dado um conjunto de treinamento $(\mathbf{x}_i, y_i)$ e uma margem fixa γ_f, um erro ocorre se $y_i(\mathbf{x}_i \cdot \boldsymbol{\beta} + b) < \gamma_f \|\boldsymbol{\beta}\|$, sendo $\boldsymbol{\beta}$ o vetor normal ao hiperplano de separação e b o termo *bias*. Assim, a função de erro foi definida por [11] como mostrado na Equação 5.

$$J(\boldsymbol{\beta}) = \sum_{(\mathbf{x}_i, y_i) \in M} (\gamma_f \|\boldsymbol{\beta}\| - y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + b)), \tag{5}$$

onde M é um subgrupo dos dados de treinamento Z, tal que: $M = (\mathbf{x}_i, y_i) \in Z | y_i(\boldsymbol{\beta} \cdot \mathbf{x}_i + b) < \gamma_f \|\boldsymbol{\beta}\|$.

A partir daí, baseado na regra de atualização de pesos do perceptron e adicionado um termo relacionado a margem fixa, estabeleceu-se a regra de atualização mostrada na Equação 6.

$$\begin{aligned} &\boldsymbol{\beta}^{t+1} \leftarrow \boldsymbol{\beta}^t \lambda_t + \eta y_i \mathbf{x}_i \text{ tal que :} \\ &\lambda_t = \begin{cases} 1 - \frac{\eta \gamma_f}{\|\boldsymbol{\beta}^t\|} & \text{se } \|\boldsymbol{\beta}^t\| \neq 0 \\ 1 & \text{caso contrário,} \end{cases} \\ &b^{t+1} \leftarrow b^t + \eta y_i, \end{aligned} \tag{6}$$

tal que $\eta > 0$ é a taxa de aprendizado.

Uma limitação do algoritmo FMP, definido em variáveis primais, é que ele é aplicável apenas a problemas linearmente separáveis. Para lidar com isso, o trabalho [20] propôs adaptar e usar a solução de margem flexível introduzida em [17]. Com essa proposta, uma variável de folga, representada por $\alpha_i \lambda$, é usada em cada verificação de viabilidade [20]:

$$y_i(\boldsymbol{\beta}^t \cdot \mathbf{x}_i + b) \geq \gamma_f \|\boldsymbol{\beta}\|_q - \alpha_i \lambda, \tag{7}$$

tal que λ é um hiper-parâmetro e α_i é o i-ésimo componente do vetor de multiplicadores α associado à amostra $(\mathbf{x}_i, y_i)$. Assim como $\boldsymbol{\beta}$, α_i é atualizado toda vez que ocorre um erro:

$$\begin{aligned} &\alpha^{t+1} \leftarrow \alpha^t (1 - \frac{\eta \gamma_f}{\|\boldsymbol{\beta}\|}), \\ &\alpha_i^{t+1} \leftarrow \alpha_i^{t+1} + \eta \cdot 1. \end{aligned} \tag{8}$$

3.3 Algoritmo de Margem Incremental

A proposta básica do IMA é executar o FMP sucessivamente, aumentando o tamanho da margem em cada iteração. O algoritmo permanece neste *loop* até que o FMP falhe em convergir em um número máximo de iterações ou atingir um número desejado de execuções do algoritmo FMP.

A formulação para o problema de maximização de margem, proposta por [11], é desenvolvida a partir da observação de que, uma vez obtida a margem máxima, pontos ou vetores de suporte de classes opostas estão à mesma distância do hiperplano separador [21].

A partir dessa observação foi desenvolvida a regra para atualização da margem mostrada na Equação 9.

$$\begin{aligned} &\gamma_f = max((\gamma^+(\boldsymbol{\beta}) + \gamma^-(\boldsymbol{\beta}))/2, (1+\delta)\gamma_f), \\ &\gamma^+(\boldsymbol{\beta}) = min\{y_i(\mathbf{x}_i \cdot \boldsymbol{\beta} + b)/\|\boldsymbol{\beta}\|, \forall y_i = +1\}, \\ &\gamma^-(\boldsymbol{\beta}) = min\{y_i(\mathbf{x}_i \cdot \boldsymbol{\beta} + b)/\|\boldsymbol{\beta}\|, \forall y_i = -1\}, \end{aligned} \tag{9}$$

sendo que $\delta \in (0, 1)$ é um incremento de margem mínimo definido como um hiperparâmetro. Com esta regra de atualização, garante-se que o valor da margem fixa passado ao FMP seja sempre maior que o anterior.

4 MÉTODO PROPOSTO

Neste trabalho propõe-se um algoritmo de aprendizado de margem larga para SLFNs, denominado RP-IMA. Esse método consiste em basicamente duas etapas:

(1) Execute as duas primeiras etapas do treinamento da ELM conforme descrito na seção anterior: atribua aleatoriamente os pesos e o termo *bias* da camada escondida e calcule a matriz de mapeamento **H**.
(2) Aplique o algoritmo de margem incremental (IMA) com o esquema de margem flexível, usando o conjunto (**H**, **Y**) como dados de entrada. O $\boldsymbol{\beta}$ final, retornado do IMA, será o vetor de pesos de saída $\boldsymbol{\beta}$ do RP-IMA.

O Algoritmo 1 abaixo mostra o pseudocódigo do RP-IMA implementado nesse trabalho.

Algoritmo 1 RP-IMA

1: **Hiperparâmetros**: número de neurônios da camada escondida (Ñ), passo de atualização utilizado pelo FMP (η), incremento mínimo da margem a cada iteração do IMA (δ), termo constante da variável de folga (λ), número de atualizações máximo (T_MAX).
2: **Entrada:** *dataset* de treinamento com N amostras $\{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathbf{R}^n, y_i \in \{-1, 1\}, i = 1, ..., N\}$
3: **Saída:** Rede neural com pesos definidos.
4: Gerar aleatoriamente os pesos de entrada $\mathbf{w}_i$ e o termo *bias* b_i, i = 1, ..., Ñ, sendo Ñ o número de neurônios da camada escondida.
5: Calcular a matriz de mapeamento **H**,
6: $\boldsymbol{\beta} \leftarrow 0; \gamma_f \leftarrow 0$
7: **while** FMP convergir em no máximo T_MAX atualizações **do**
8: $\quad \boldsymbol{\beta} \leftarrow \text{FMP}((\mathbf{H}, \mathbf{y}), \boldsymbol{\beta}, \gamma_f, \eta, \text{T\_MAX})$
9: $\quad \gamma_f = max((\gamma^+(\boldsymbol{\beta}) + \gamma^-(\boldsymbol{\beta}))/2, (1+\delta)\gamma_f)$
10: **end while**
11: **return:** pesos aprendidos da rede neural.

5 METODOLOGIA DOS EXPERIMENTOS

Esta seção visa explicar como os experimentos do artigo foram realizados. Como será visto a seguir, foram realizados testes com bases de dados tanto geradas de forma sintética quanto obtidas de um repositório de aprendizado de máquina.

5.1 Experimento com bases sintéticas

Inicialmente foram utilizadas bases sintéticas de 2 dimensões, com o objetivo de se verificar visualmente a superfície de separação gerada por um modelo RP-IMA comparada a gerada por um modelo ELM. Quatro *datasets* foram avaliados neste experimento, cada um com 1000 amostras (500 de cada classe).

Para reduzir o efeito aleatório inerente da ELM e RP-IMA na avaliação dos modelos, ambos os métodos utilizaram os mesmos pesos aleatórios dos neurônios ocultos. Ou seja, em cada teste realizado, os pesos ocultos gerados aleatoriamente para a ELM foram replicados para o RP-IMA. Assim, os dois modelos possuem a mesma matriz H. Para esses testes, foram gerados modelos utilizando 100 neurônios na camada oculta, tendo como função de ativação a tangente hiperbólica.

Os outros hiperparâmetros usados para o RP-IMA foram: $\eta = 0.1$, $\lambda = 0.1$ e $\delta = 10^{-3}$. Também, como critério de parada do algoritmo, foi definido um máximo de 10.000 atualizações do vetor $\boldsymbol{\beta}$ e permitiu-se a execução de no máximo 20 iterações do IMA.

Além do resultado visual da superfície de separação, os modelos também foram avaliados quanto a acurácia e o tamanho da margem de separação obtida. Esses resultados estão expressos em tabelas, que apresentam o valor médio obtido de uma validação cruzada com 10 dobras. Os valores entre parênteses nas tabelas representam os respectivos desvios padrão.

Os valores de tamanho da margem mostrados neste artigo são na verdade correspondentes a margem geométrica γ, que é definida por:

$$\gamma = \min_{i=1,\ldots,N}(\gamma_i), \text{ tal que } \gamma_i = y_i(\boldsymbol{\beta}^t \mathbf{x}_i + b)/\|\boldsymbol{\beta}\| \tag{10}$$

Este valor foi calculado no espaço de características, ou seja, usando os vetores linha $\mathbf{h}_i$ da matriz $\mathbf{H}$. Desta forma, tem-se:

$$\gamma = \min_{i=1,\ldots,N}(\gamma_i), \text{ tal que } \gamma_i = y_i(\boldsymbol{\beta}^t \mathbf{h}_i + b)/\|\boldsymbol{\beta}\|, \tag{11}$$

sendo $\boldsymbol{\beta}$ o vetor de pesos da camada de saída da ELM ou RP-IMA.

5.2 Experimento com bases de *benchmark*

Os conjuntos de dados usados neste experimento foram obtidos do repositório de aprendizado de máquina da UCI [5], e estão apresentados na Tabela 1. Alguns problemas, como o Iris, foram readaptados para ter apenas duas classes. A tabela 1 detalha informações básicas sobre os conjuntos de dados utilizados, como número de características (f), quantidade de amostras de cada classe e total de amostras.

Tabela 1: *Datasets* utilizados no experimento.

Dataset	Sigla	f	Amostras		
			+1	-1	Total
Iris	IRI	4	50	100	150
Synthetic	SYN	60	300	300	600
Mushroom	MUS	22	2156	3488	5644
Banknote	BAN	4	610	762	1372
Ionosphere	ION	34	225	126	351
Spam	SPA	57	1813	2788	4601
Diabetes	DIA	8	268	500	768

Neste experimento, modelos RP-IMA foram comparados a modelos ELM e SVM utilizando *kernel* RBF. Todos os conjuntos de dados foram normalizados para o intervalo [0, 1] antes do treinamento e os hiperparâmetros do RP-IMA utilizados foram: $\eta = 0.1$, $\delta = 10^{-3}$ e T_MAX = 10.000. Além disso, permitiu-se a operação de no máximo 20 execuções do FMP. Para selecionar os hiperparâmetros λ do RP-IMA e o parâmetro de regularização C do SVM utilizou-se um *grid search cross-validation*. O coeficiente do *kernel* γ do SVM foi definido como $\gamma = 1/(n_{features} \cdot var(X))$, onde $n_{features}$ é o número de características das entradas X e $var(X)$ é a variância de X. Para cada conjunto de dados, testou-se a ELM e RP-IMA com três quantidades diferentes de neurônios ocultos, dadas por N, N/2 e N/3, onde N é igual ao total de amostras em cada conjunto de dados, limitado por 1000. Em todos os experimentos, a tangente hiperbólica foi utilizada como função de ativação dos neurônios ocultos. Os pesos aleatórios dos neurônios da camada oculta selecionados para a ELM e para o RP-IMA foram os mesmos.

Assim como no experimento anterior, os resultados de acurácia e tamanho da margem de separação obtida estão expressos em tabelas, que apresentam o valor médio obtido de uma validação cruzada com 10 dobras. Os valores entre parênteses nas tabelas representam os respectivos desvios padrão.

6 RESULTADOS E DISCUSSÕES

Esta seção tem como objetivo apresentar e analisar os resultados obtidos dos experimentos considerando bases de dados sintéticas e reais.

6.1 Experimento com bases sintéticas

A Figura 2 mostra as bordas de decisão obtidas a partir dos dois modelos (ELM à esquerda e RP-IMA à direita) para cada um dos conjuntos de dados gerados.

Ao se analisar a Figura 2, nota-se que, em vários pontos, os modelos RP-IMA resultaram em melhores margens de separação entre as classes. Além disso, enquanto os modelos ELM apresentaram limites de decisão com sinais claros de *overfitting*, os modelos RP-IMA geraram superfícies de separação mais suaves.

A Tabela 2 apresenta os resultados de acurácia e tamanho da margem obtidos para o RP-IMA e ELM. Devido a utilização de bases sintéticas simples já era esperado ambos os modelos alcançarem resultados de acurácia estatisticamente similares e próximos de 100%. Assim, o resultado mais importante aqui é observar que, em todos os casos, os modelos RP-IMA resultaram em uma margem de separação significativamente maior que as geradas por modelos ELM.

Tabela 2: Resultados obtidos para as bases sintéticas.

Algoritmo	Acurácia	Margem Média
Gaussian blobs		
ELM	99.5 (0.5)	$5.0 \cdot 10^{-8}$ ($4.9 \cdot 10^{-8}$)
RP-IMA	100.0 (0.0)	0.5796 (0.1006)
Xor		
ELM	99.7 (0.6)	$1.7 \cdot 10^{-7}$ ($1.4 \cdot 10^{-7}$)
RP-IMA	99.7 (0.6)	0.0673 (0.0109)
Circles		
ELM	99.8 (0.6)	$8.3 \cdot 10^{-13}$ ($5.3 \cdot 10^{-13}$)
RP-IMA	100.0 (0.0)	0.0254 (0.0030)
Spirals		
ELM	99.6 (0.7)	0.0003 (0.0003)
RP-IMA	99.8 (0.4)	0.0274 (0.0069)

6.2 Experimento com bases de *benchmark*

A Tabela 3 mostra os resultados de acurácia obtidos para o RP-IMA, ELM e SVM. Primeiramente é importante observar que, comparado

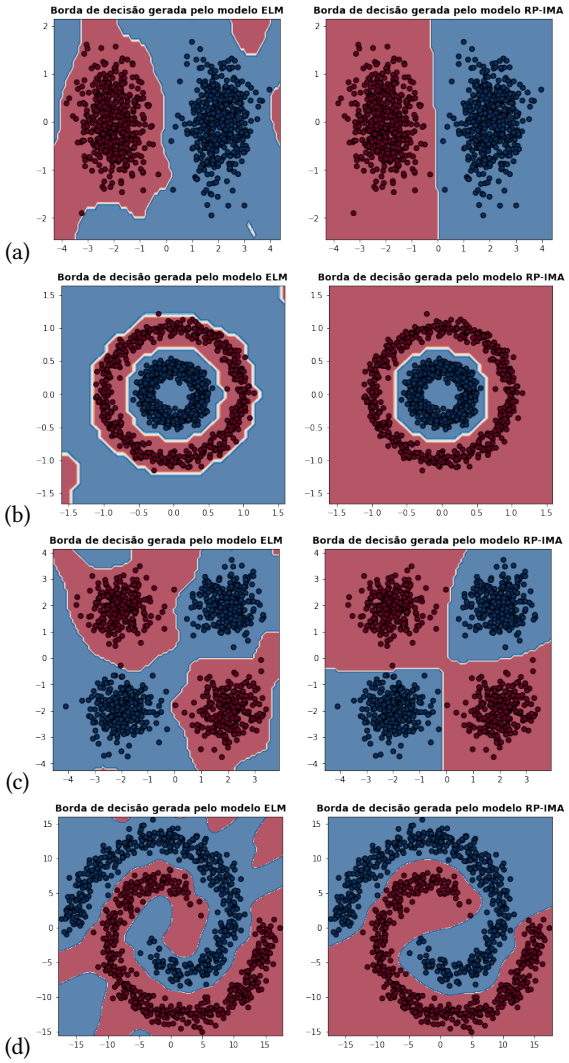

Figura 2: Superfície de classificação gerada pela ELM (à esquerda) e pelo RP-IMA (à direita). *Datasets*: (a) Gaussian blobs; (b) Circles; (c) Xor; (d) Spirals.

à ELM, o RP-IMA obteve menor variação de performance de acordo com a quantidade de neurônios da camada escondida. Isso foi considerado pelos autores como mais um indício que o RP-IMA está menos propenso a *overfitting* que a ELM.

Além disso, observa-se da Tabela 3 que, juntamente com a SVM, o RP-IMA com N neurônios na camada escondida foi o método que obteve maior acurácia média em mais testes, 4 ocasiões. Por fim, nota-se também que, com exceção da base de dados *Mushroom*, em todas as outras o RP-IMA obteve resultados de acurácia iguais ou superiores ELM, para todas as quantidades de neurônios testadas.

A Tabela 4 mostra os resultados de tamanho da margem obtidos pelos modelos RP-IMA e ELM, para os casos em que ambos os modelo obtiveram acurácia de treinamento igual a 100%. Nota-se que, em todos os conjuntos de dados e considerando qualquer número de neurônios testados, o RP-IMA resultou em valores de margem significativamente maiores do que os obtidos com o método ELM convencional.

Para as bases de dados da Tabela 4 foram gerados os gráficos da Figura 3 que mostram a evolução do tamanho da margem de acordo com o número de execuções do FMP. Cada ponto dos gráficos representa o valor médio da margem obtido em cada iteração do IMA, e as barras de erro representam o erro padrão.

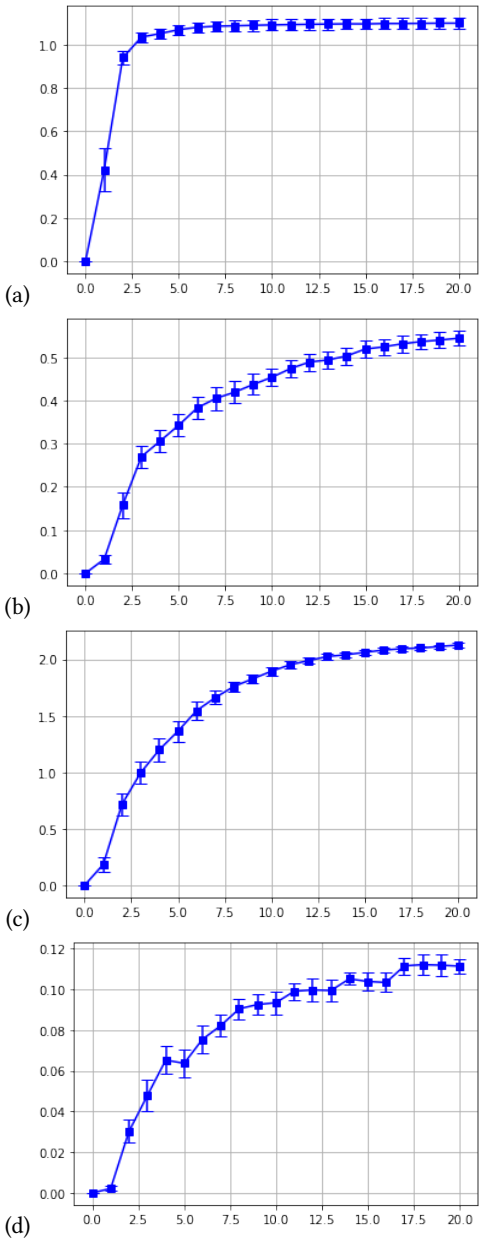

Figura 3: Gráficos da evolução da margem para cada iteração do RP-IMA. No eixo horizontal está representado as iterações do RP-IMA e no eixo vertical tem-se o valor da margem geométrica calculada. *Datasets*: (a) Iris; (b) Synthetic control chart time series; (c) Mushroom; (d) Banknote.

Tabela 3: Valores de acurácia média obtidos da execução de uma validação cruzada com 10 dobras.

Set	N	RP-IMA			ELM			SVM
		N neurônios	N/2 neurônios	N/3 neurônios	N neurônios	N/2 neurônios	N/3 neurônios	
IRI	150	**100.00 (0.00)**	**100.00 (0.00)**	**100.00 (0.00)**	99.33 (2.00)	**100.00 (0.00)**	**100.00 (0.00)**	**100.00 (0.00)**
SYN	600	99.50 (0.76)	98.83 (1.83)	98.50 (2.03)	67.50 (6.72)	94.59 (3.80)	97.83 (1.83)	**100.00 (0.00)**
MUS	1000	99.98 (0.05)	99.98 (0.05)	**100.00 (0.00)**	**100.00 (0.00)**	**100.00 (0.00)**	**100.00 (0.00)**	99.72 (0.16)
BAN	1000	**100.00 (0.00)**	**100.00 (0.00)**	**100.00 (0.00)**	93.80 (2.76)	99.56 (0.48)	**100.00 (0.00)**	**100.00 (0.00)**
ION	351	94.01 (3.93)	91.44 (5.58)	93.15 (4.09)	73.22 (4.79)	83.77 (4.57)	86.61 (6.39)	**95.16 (3.84)**
SPA	1000	**93.47 (0.93)**	92.33 (3.09)	91.83 (2.72)	89.55 (1.89)	91.09 (2.94)	91.23 (2.55)	93.15 (2.17)
DIA	768	**77.61 (2.37)**	74.87 (3.31)	75.78 (3.62)	55.05 (4.95)	67.07 (4.77)	72.27 (3.72)	77.47 (3.74)

Tabela 4: Valores de margem médios obtidos da execução de uma validação cruzada com 10 dobras.

Set	RP-IMA			ELM		
	N neurônios	N/2 neurônios	N/3 neurônios	N neurônios	N/2 neurônios	N/3 neurônios
IRI	1.1003 (0.0723)	0.8330 (0.1013)	0.6467 (0.0707)	$1.9 \cdot 10^{-5}$ ($4.9 \cdot 10^{-6}$)	0.0002 (0.0001)	0.0008 (0.0002)
SYN	0.5442 (0.0507)	0.2327 (0.0481)	0.1179 (0.0476)	0.0123 (0.0019)	0.0020 (0.0017)	0.0012 (0.0014)
MUS	2.1258 (0.0575)	1.2434 (0.1036)	0.8499 (0.0851)	0.7108 (0.2945)	0.5683 (0.2226)	0.2860 (0.1405)
BAN	0.1413 (0.0053)	0.1008 (0.0043)	0.0812 (0.0055)	$7.2 \cdot 10^{-12}$ ($6.2 \cdot 10^{-6}$)	$1.9 \cdot 10^{-9}$ ($3.8 \cdot 10^{-10}$)	$2.4 \cdot 10^{-8}$ ($4.3 \cdot 10^{-9}$)

Devido ao esquema de margem flexível não é possível garantir que o tamanho da margem será sempre maior a cada iteração do RP-IMA. Contudo, nos gráficos da Figura 3 é possível observar um crescimento consistente da margem, principalmente nas primeiras iterações do algoritmo.

7 CONCLUSÃO

Esse artigo buscou explorar o conceito de mapeamento explícito aleatório com funções de ativação. Foi apresentado um novo algoritmo de treinamento para SLFN com projeção aleatória e margem larga baseado na definição primal do IMA.

Como pode ser observado nos resultados, o método proposto, RP-IMA, obteve soluções de maior margem em relação ao *baseline* (ELM) sem maximização de margem. Além disso, os gráficos da evolução da margem para cada iteração do RP-IMA (Figura 3) mostraram um crescimento consistente da margem, com um aumento significativo da mesma durante as primeiras iterações do algoritmo.

Por fim, pôde-se observar que o desempenho do novo método é menos sensível a variações na arquitetura da rede. Na maioria dos testes de acurácia, o RP-IMA teve um desempenho melhor que a ELM e, assim como a SVM, obteve maior acurácia média em 4 das 7 bases de dados de *benchmark* testadas.

ACKNOWLEDGMENTS

Os autores agradecem à Pró-Reitoria de Pesquisa (PRPq) da Universidade Federal de Minas Gerais (UFMG), FAPEMIG, CNPq e CAPES pelo apoio financeiro.

REFERÊNCIAS

[1] Janier Arias-Garcia, Augusto Mafra, Liliane Gade, Frederico Coelho, Cristiano Castro, Luiz Torres, and Antônio Braga. 2020. Enhancing performance of gabriel graph-based classifiers by a hardware co-processor for embedded system applications. *IEEE Transactions on Industrial Informatics* 17, 2 (2020), 1186–1196.

[2] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*. 144–152.

[3] T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13, 1 (1967), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[4] C. Domeniconi, D. Gunopulos, and Jing Peng. 2005. Large margin nearest neighbor classifiers. *IEEE Transactions on Neural Networks* 16, 4 (2005), 899–909. https://doi.org/10.1109/TNN.2005.849821

[5] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[6] Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. 2018. Large margin deep networks for classification. *Advances in neural information processing systems* 31 (2018).

[7] K Ruben Gabriel and Robert R Sokal. 1969. A new statistical approach to geographic variation analysis. *Systematic zoology* 18, 3 (1969), 259–278.

[8] Claudio Gentile. 2000. A New Approximate Maximal Margin Classification Algorithm. In *Advances in Neural Information Processing Systems*, T. Leen, T. Dietterich, and V. Tresp (Eds.), Vol. 13. MIT Press. https://proceedings.neurips.cc/paper/2000/file/d072677d210ac4c03ba046120f0802ec-Paper.pdf

[9] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. 2004. Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, Vol. 2. 985–990 vol.2. https://doi.org/10.1109/IJCNN.2004.1380068

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

[11] Saul C Leite and Raul Fonseca Neto. 2008. Incremental margin algorithm for large margin classifiers. *Neurocomputing* 71 (2008), 1550–1560.

[12] Tiago Matias, Francisco Souza, Rui Araújo, and Carlos Henggeler Antunes. 2014. Learning of a single-hidden layer feedforward neural network using an optimized extreme learning machine. *Neurocomputing* 129 (2014), 428–436.

[13] Yoh-Han Pao, Gwang-Hoon Park, and Dejan J. Sobajic. 1994. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing* 6, 2 (1994), 163–180. https://doi.org/10.1016/0925-2312(94)90053-1 Backpropagation, Part IV.

[14] Y.-H. Pao and Y. Takefuji. 1992. Functional-link net computing: theory, system architecture, and functionalities. *Computer* 25, 5 (1992), 76–79. https://doi.org/10.1109/2.144401

[15] Frank Rosenblatt. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65, 6 (1958), 386.

[16] Wouter F Schmidt, Martin A Kraaijveld, Robert PW Duin, et al. 1992. Feed forward neural networks with random weights. In *International conference on pattern recognition*. IEEE Computer Society Press, 1–1.

[17] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press.

[18] LCB Torres, CL Castro, F Coelho, F Sill Torres, and AP Braga. 2015. Distance-based large margin classifier suitable for integrated circuit implementation. *Electronics Letters* 51, 24 (2015), 1967–1969.

[19] Luiz CB Torres, Cristiano L Castro, and Antônio P Braga. 2015. Gabriel graph for dataset structure and large margin classification: A Bayesian approach. In *Proceedings of the European Symposium on Neural Networks*. 237–242.

[20] Saulo Moraes Villela, Saul de Castro Leite, and Raul Fonseca Neto. 2016. Incremental p-margin algorithm for classification with arbitrary norm. *Pattern Recognition* 55 (2016), 261–272.

[21] Saulo Moraes Villela, SC Leite, and R Fonseca Neto. 2013. Algoritmo de margem incremental com norma p para classificadores de larga margem. In *XI CBIC-Congresso Brasileiro de Inteligência Computacional, Porto de Galinhas, PE.*

[22] Kilian Q Weinberger and Lawrence K Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research* 10, 2 (2009).

[23] Shi-Xiong Zhang, Chaojun Liu, Kaisheng Yao, and Yifan Gong. 2015. Deep neural support vector machines for speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4275–4279. https://doi.org/10.1109/ICASSP.2015.7178777