Máquina de Aprendizado Extremo Robusta para Classificação com Outliers

Ana Luiza B. P. Barros † ‡
† Departamento de Ciência da Computação
Universidade Estadual do Ceará
Campus do Itaperi
Fortaleza, Ceará, Brasil
Email: analuiza@larces.uece.br

Guilherme A. Barreto ‡
‡ Departamento de Engenharia de Teleinformática
Universidade Federal do Ceará
Centro de Tecnologia
Campus do Pici
Fortaleza, Ceará, Brasil
Email: guilherme@deti.ufc.br

***Resumo*—A Máquina de Aprendizado Extremo (ELM - Extreme Learning Machine), recentemente proposta por Huang *et al*. [6], é uma arquitetura de rede neural com uma única camada oculta, que tem sido aplicada com sucesso a tarefas de classificação e regressão não-linear [5]. Um passo fundamental no projeto da ELM é o cálculo da matriz de pesos de saída, um passo usualmente realizado por meio do método mínimos quadrados ordinários (OLS - Ordinary Least-Squares), também conhecido como a técnica da inversa generalizada Moore-Penrose. A partir da teoria de regressão robusta, é bem conhecido que o método OLS produz modelos preditivos altamente sensíveis a outliers nos dados. Nesse artigo, desenvolvemos uma extensão da ELM que é robusta a outliers causados por erros na rotulação dos dados. Para lidar com esse problema, sugerimos o uso de estimadores-M, um framework de estimação de parâmetros largamente utilizado em regressão robusta, para calcular a matriz de pesos de saída, em vez de usar a solução OLS padrão. O modelo proposto é robusto a ruído no rótulo não apenas próximo às fronteiras das classes, mas também distante dessa fronteiras, que podem resultar em erros na rotulação ou erros grosseiros na medição das características de entrada. Mostramos a utilidade da abordagem proposta de classificação através de resultados de simulação usando dados sintéticos e do mundo real.**

I. Introdução

Nos últimos anos, tem havido um interesse cada vez maior em uma classe de modelo de rede neural supervisionado, com uma única camada oculta, genericamente chamado Máquina de Apredizado Extremo (ELM - Extreme Learning Machine), na qual os pesos entre as camadas de entrada e oculta são escolhidos aleatoriamente, e entre as camadas oculta e de saída, são determinados analiticamente. Devido principalmente à sua rapidez na aprendizagem e facilidade de implementação [5], vários autores têm aplicado a rede ELM padrão (e sofisticadas variações suas) a um número de problemas complexos em classificação de padrões e regressão [1], [4], [13]–[18].

Os trabalhos acima mencionados não têm abordado questões importantes de desempenho do modelo na presença de outliers nos dados, com o trabalho de Horata *et al.* [4] sendo a única exceção. Na verdade, nos últimos anos, tem-se observado um interesse crescente no desenvolvimento de arquiteturas de redes neurais que são robustas a outliers, incluindo propostas para o projeto de redes RBF [10], [11], redes echo-state [12] e até mesmo redes ELM [4].

Vale ressaltar que todos os trabalhos anteriores (sem exceção!) abordaram a questão da robustez a outliers para problemas de regressão, tais como aproximação de função e predição de séries temporais. No entanto, em muitos problemas de classificação de padrões do mundo real, os rótulos providos para as amostras de dados são ruidosos. Existem tipicamente dois tipos de ruído em rótulos. Ruído próximo às fronteiras das classes, que muitas vezes ocorrem porque é difícil rotular de forma consistente pontos de dados ambíguos. Erros de rotulação longe das fronteiras das classes, que podem ocorrer por causa de enganos na rotulação ou erros grosseiros na medição das características de entrada. Erros de rotulação longe da fronteira compreendem uma categoria particular de *outliers* [9].

Na verdade, não fomos capazes de encontrar um único artigo que avaliasse a robustez de modelos de redes neurais, especialmente as redes baseadas em ELM, em problemas de classificação na presença de outliers. Portanto, a fim de permitir que classificadores baseados em ELM tratem eficientemente erros de rotulação, nesse artigo propomos o uso de estimadores-M [8], um framework largamente utilizado para estimação de parâmetros em problemas de regressão robusta, para calcular o operador da matriz pesos, em vez de usar a solução de mínimos quadrados ordinários. Mostramos através de simulações em dados sintéticos e do mundo real, que o classificador ELM resultante é muito robusto a esse tipo de outliers. Até onde sabemos, essa é a primeira vez que o desempenho de classificação da rede ELM é avaliado sob a presença de outliers.

O restante do artigo é organizado como segue. Na Seção II, revisamos rapidamente os fundamentos de ELM no contexto de classificação de padrões. Em seguida, na Seção III, descrevemos as ideias e conceitos básicos por trás do framework de estimação-M e introduzimos nossa abordagem sobre classificação robusta e supervisionada de padrões, usando ELM. Na Seção IV apresentamos os experimentos computacionais realizados com o uso de conjuntos de dados sintéticos e reais, e também discutimos os resultados obtidos. O artigo é concluído na Seção V.

II. FUNDAMENTOS DE ELM

Suponhamos que N pares de dados $\{(\mathbf{x}_\mu, \mathbf{d}_\mu)\}_{\mu=1}^{N}$ estão disponíveis para a construção e avaliação do modelo, onde $\mathbf{x}_\mu \in \mathbb{R}^{p+1}$ é o μ-ésimo padrão de entrada[1] e $\mathbf{d}_\mu \in \mathbb{R}^K$ é o rótulo da classe alvo correspondente, com K denotando o número de classes. Para os rótulos, assumimos um esquema de codificação 1-of-K, isto é, para cada vetor de rótulos $\mathbf{d}_\mu$, o componente cujo índice corresponde à classe do padrão $\mathbf{x}_\mu$ é setado para "+1", enquanto os outros $K-1$ componentes são setados para "-1".

Então, selecionemos aleatoriamente N_1 ($N_1 < N$) pares de dados a partir do conjunto de dados disponível e organizemo-os ao longo das colunas das matrizes $\mathbf{D}$ e $\mathbf{X}$, como segue:

$$\mathbf{X} = [\mathbf{x}_1 \mid \mathbf{x}_2 \mid \cdots \mid \mathbf{x}_{N_1}] \quad \text{and} \quad \mathbf{D} = [\mathbf{d}_1 \mid \mathbf{d}_2 \mid \cdots \mid \mathbf{d}_{N_1}]. \tag{1}$$

onde $\dim(\mathbf{X}) = (p+1) \times N_1$ e $\dim(\mathbf{D}) = m \times N_1$.

ELM é uma rede feedforward com uma única camada oculta (SLFN - Single-hidden Layer Feedforward Network), proposta por [6], para a qual os pesos das entradas para os neurônios ocultos são escolhidos aleatoriamente, enquanto apenas os pesos dos neurônios ocultos para a saída são analiticamente determinados. Consequentemente, ELM oferece vantagens significativas, tais como rapidez na aprendizagem, facilidade de implementação, e menos intervenção humana quando comparada a SLFNs mais tradicionais, como as redes MLP e RBF. Para uma rede com p unidades de entrada, q neurônios ocultos e C saídas, a i-ésima saída no passo k, é dada por

$$o_i(k) = \boldsymbol{\beta}_i^T \mathbf{h}(k), \tag{2}$$

onde $\boldsymbol{\beta}_i \in \mathbb{R}^q$, $i = 1, \ldots, C$, o vetor de pesos que conecta os neurônios ocultos ao i-ésimo neurônio de saída, e $\mathbf{h}(k) \in \mathbb{R}^q$ é o vetor de saídas dos neurônios ocultos para um dado padrão de entrada $\mathbf{x}(k) \in \mathbb{R}^p$. O vetor $\mathbf{h}(k)$, propriamente dito, é definido como

$$\mathbf{h}(k) = [f(\mathbf{w}_1^T \mathbf{x}(k) + b_1), \ldots, f(\mathbf{w}_q^T \mathbf{x}(k) + b_q)]^T, \tag{3}$$

onde b_l, $l = 1, \ldots, q$, é o bias do l-ésimo neurônio oculto, $\mathbf{w}_l \in \mathbb{R}^p$ é o vetor de pesos do l-ésimo neurônio oculto e $f(\cdot)$ é uma função de ativação sigmoidal. Usualmente, os vetores de pesos $\mathbf{w}_l$ são amostrados aleatoriamente a partir de um distribuição uniforme (ou normal).

Seja $\mathbf{H} = [\mathbf{h}(1)\ \mathbf{h}(2)\ \cdots\ \mathbf{h}(N)]$ uma matriz $q \times N$ cujas N colunas são os vetores de saída da camada oculta $\mathbf{h}(k) \in \mathbb{R}^q$, $k = 1, ..., N$, onde N é o número de padrões de entrada disponíveis para treinamento. Similarmente, seja $\mathbf{D} = [\mathbf{d}(1)\ \mathbf{d}(2)\ \cdots\ \mathbf{d}(N)]$ uma matriz $C \times N$ cuja k-ésima coluna é o vetor alvo (desejado) $\mathbf{d}(k) \in \mathbb{R}^C$ associado com o padrão de entrada $\mathbf{x}(k)$, $k = 1, ..., N$.

Finalmente, seja $\boldsymbol{\beta} = [\boldsymbol{\beta}_1\ \boldsymbol{\beta}_2\ \cdots\ \boldsymbol{\beta}_C]$ uma matriz $q \times C$, cuja i-ésima coluna é o vetor peso $\boldsymbol{\beta}_i \in \mathbb{R}^q$, $i = 1, ..., C$. Assim, essas três matrizes estão relacionadas pelo seguinte mapeamento linear:

$$\mathbf{D} = \boldsymbol{\beta}^T \mathbf{H}, \tag{4}$$

onde as matrizes $\mathbf{D}$ e $\mathbf{H}$ são conhecidas, enquanto que a matriz peso $\boldsymbol{\beta}$ não é. A solução OLS do sistema linear da Eq. (4) é dada pela inversa generalizada Moore-Penrose, como segue:

$$\boldsymbol{\beta} = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{D}^T. \tag{5}$$

Eq. (5) pode ser dividia em C equações de estimação individuais, uma para cada neurônio de saída i, sendo escrita como

$$\boldsymbol{\beta}_i = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{D}_i^T, \qquad i = 1, \ldots, C, \tag{6}$$

onde $\mathbf{D}_i$ denota a i-ésima linha da matriz $\mathbf{D}$.

Em vários problemas do mundo real, a matriz $\mathbf{H}\mathbf{H}^T$ pode ser singular, prejudicando o uso da Eq. (5). Na verdade, uma matriz $\mathbf{H}\mathbf{H}^T$ próximo à singular (ainda inversível) é também um problema, pois ela pode levar a resultados numericamente instáveis. Para evitar ambos os problemas, uma abordagem comum envolve o uso do método ridge regression (também conhecido como regularização Tikhonov), que é dado por

$$\boldsymbol{\beta}_i = (\mathbf{H}\mathbf{H}^T + \lambda \mathbf{I})^{-1} \mathbf{H}\mathbf{D}_i^T, \qquad i = 1, \ldots, C, \tag{7}$$

onde a constante $\lambda > 0$ é o parâmetro de regularização.

Como mencionado na introdução, no que diz respeito à robustez da ELM a outliers em problemas de classificação, até onde sabemos, ainda falta uma abordagem compreensiva.Tendo isso em mente, propomos o uso de técnicas de regressão robusta para calcular a matriz de pesos da saída, no lugar da abordagem OLS. A abordagem proposta é descrita na próxima seção.

III. NOÇÕES BÁSICAS DE ESTIMAÇÃO-M

Uma importante característica do OLS é que ele atribui a mesma importância a todas as amostras de erro, isto é, todos os erros contribuem da mesma forma para a solução final. Uma abordagem comum para tratar esse problema consiste em remover outliers dos dados e, então, tentar o ajuste usual de mínimos quadrados. Uma abordagem mais consistente, conhecida como *regressão robusta*, usa métodos de estimação não tão sensíveis a outliers, como o OLS.

Huber [7] introduziu o conceito de estimação-M, onde M significa do tipo "máxima verossimilhança", onde a robustez é obtida minimizando uma outra função, diferente da soma do erros quadráticos. Com base na teoria de Huber, um estimador-M geral, aplicado ao i-ésimo neurônio de saída do classificador ELM, minimiza a seguinte função objetivo:

$$J(\boldsymbol{\beta}_i) = \sum_{\mu=1}^{N} \rho(e_{i\mu}) = \sum_{\mu=1}^{N} \rho(d_{i\mu} - y_{i\mu}) = \sum_{\mu=1}^{N} \rho(d_{i\mu} - \boldsymbol{\beta}_i^T \mathbf{h}_\mu), \tag{8}$$

[1] A primeira componente de $\mathbf{x}_\mu$ é igual a 1, a fim de incluir o bias.

onde a função $\rho(\cdot)$ calcula a contribuição de cada erro $e_{i\mu} = d_{i\mu} - y_{i\mu}$ para a função objetivo, $d_{i\mu}$ é o valor alvo do i-ésimo neurônio de saída para o μ-ésimo padrão de entrada $\mathbf{x}_\mu$, e $\boldsymbol{\beta}_i$ é o vetor de pesos do i-ésimo neurônio de saída. O OLS é um estimador-M particular, obtido quando $\rho(e_{i\mu}) = e_{i\mu}^2$. É desejável que a função ρ possua as seguintes propriedades:

Propriedade 1: $\rho(e_{i\mu}) \geq 0$.
Propriedade 2: $\rho(0) = 0$.
Propriedade 3: $\rho(e_{i\mu}) = \rho(-e_{i\mu})$.
Propriedade 4: $\rho(e_{i\mu}) \geq \rho(e_{i'\mu})$, para $|e_{i\mu}| > |e_{i'\mu}|$.

Estimação de parâmetros é definida pela equação de estimação, que é uma função poderada da derivada da função objetivo. Seja $\psi = \rho'$ a derivada de ρ. Derivando ρ em relação ao vetor peso estimado $\hat{\boldsymbol{\beta}}_i$, temos

$$\sum_{\mu=1}^{N} \psi(y_{i\mu} - \hat{\boldsymbol{\beta}}_i^T \mathbf{x}_\mu)\mathbf{x}_\mu^T = \mathbf{0}, \tag{9}$$

onde $\mathbf{0}$ é um vetor linha de zeros com dimensão $(p+1)$. Então, definindo a função peso $w(e_{i\mu}) = \psi(e_{i\mu})/e_{i\mu}$, e fazendo $w_{i\mu} = w(e_{i\mu})$, as equações de estimação são dadas por

$$\sum_{\mu=1}^{n} w_{i\mu}(y_{i\mu} - \hat{\boldsymbol{\beta}}_i^T \mathbf{x}_\mu)\mathbf{x}_\mu^T = \mathbf{0}. \tag{10}$$

Assim, solucionar as equações de estimação corresponde a solucionar um problema ponderado de mínimos quadrados, minimizando $\sum_\mu w_{i\mu}^2 e_{i\mu}^2$.

É interessante notar, no entanto, que os pesos dependem dos resíduos (isto é, dos erros estimados), os resíduos dependem dos coeficientes estimados, e os coeficientes estimados dependem dos pesos. Como uma consequência, um método iterativo de estimação chamado *iteratively reweighted least-squares* (IRLS) [2], é comumente usado. Os passos do algoritmo IRLS, no contexto de treinamento de um classificador ELM, usando Eq. (6) como referência, são descritos a seguir.

Algoritmo IRLS para Treinamento ELM

Passo 1 - Prover uma estimativa inicial $\hat{\boldsymbol{\beta}}_i(0)$ usando a solução OLS em Eq. (6).

Passo 2 - Em cada iteração t, calcular os resíduos a partir das iterações anteriores $e_{i\mu}(t-1)$, $\mu = 1, \ldots, N$, associados com o i-ésimo neurônio de saída, e então calcular os pesos correspondentes $w_{i\mu}(t-1) = w[e_{i\mu}(t-1)]$.

Passo 3 - Resolver a nova estimativa de mínimos quadrados ponderados de $\boldsymbol{\beta}_i(t)$:

$$\hat{\boldsymbol{\beta}}_i(t) = \left[\mathbf{H}\mathbf{W}(t-1)\mathbf{H}^T\right]^{-1}\mathbf{H}\mathbf{W}(t-1)\mathbf{D}_i^T, \tag{11}$$

onde $\mathbf{W}(t-1) = \text{diag}\{w_{i\mu}(t-1)\}$ é uma matriz peso $N \times N$. Repetir Passos 2 e 3 até a convergência do vetor estimado de coeficientes $\hat{\boldsymbol{\beta}}_i(t)$.

Várias funções ponderadas para os estimadores-M podem ser escolhidas, tais como a função ponderada de Huber:

$$w(e_{i\mu}) = \begin{cases} \frac{k}{|e_{i\mu}|}, & \text{se } |e_{i\mu}| > k \\ 1, & \text{caso contrário.} \end{cases} \tag{12}$$

onde o parâmetro k é uma constante de ajuste. Valores menores de k levam a uma maior resistência a outliers, mas ao custo de menor eficiência quando os erros são normalmente distribuídos. Em particular, $k = 1.345\sigma$ para a função Huber, onde σ é uma estimativa robusta do desvio padrão dos erros[2].

Em suma, a ideia básica da abordagem proposta é muito simples: substitua a estimação OLS do vetor peso $\hat{\boldsymbol{\beta}}$ do i-ésimo neurônio de saída descrito em Eq. (6), pela estimação fornecida pelo uso combinado do framework de estimação-M e do algoritmo IRLS. A partir de agora, nos referiremos à abordagem proposta como classificador *ELM Robusto* (ou ROB-ELM, do inglês, *Robust ELM*). Na próxima seção, apresentamos e discutimos os resultados obtidos pelo classificador ROB-ELM em conjuntos de dados sintéticos e reais.

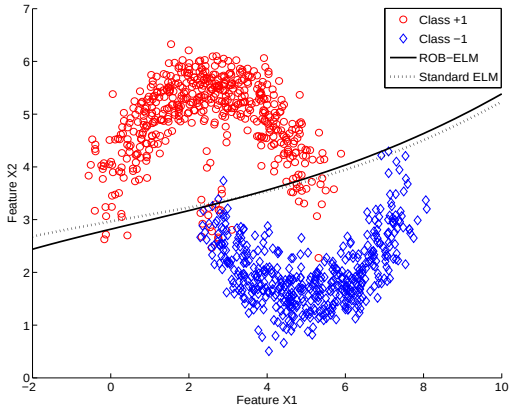
IV. Simulações e Discussão

Como uma prova de conceito, no primeiro experimento pretendemos mostrar a influência de outliers no posicionamento final da curva de decisão entre duas classes de dados não linearmente separáveis. Para esse propósito, criamos um conjunto sitético de dados, bidimensional, consistindo de $N = 120$ amostras mais N_{out} outliers. Classificadores ELM e ROB-ELM são treinados duas vezes. Na primeira vez, eles são treinados com o conjunto de dados livre de outliers. Na segunda vez, eles são treinados com os outliers adicionados ao conjunto de dados original. Vale ressaltar que todas as amostras de dados são usadas para treinar os classificadores, já que o objetivo é visualizar a posição final da curva de decisão, e não, calcular taxas de reconhecimento.

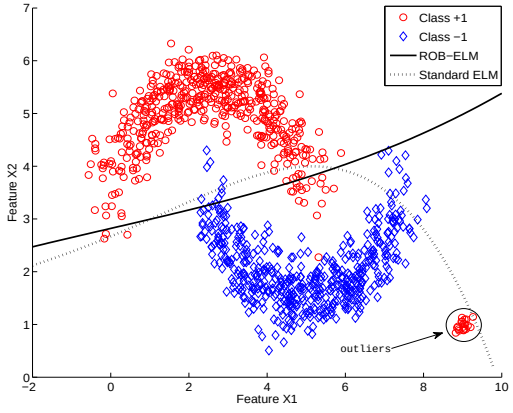
Para esse experimento, a função ponderada Andrews foi usada para implementar o classificador ROB-ELM, e a constante de regularização, necessária à implementação do classificador ELM padrão, foi setada para $\lambda = 10^{-2}$. Três neurônios ocultos com funções de ativação tangente hiperbólica foram usados para ambos os classificadores. Por uma questão de justiça, os classificadores ELM e ROB-ELM usaram os mesmos pesos entre as camadas entrada e oculta, que foram aleatoriamente amostrados a partir de uma distribuição uniforme entre $(-0.1, +0.1)$. Como parâmetro de ajuste k, foi usado o valor default da função `robustfit` do Matlab. A fim de avaliar as curvas de decisão finais dos classificadores ELM e ROB-ELM na presença de outliers, adicionamos $N_{out} = 10$ outliers ao conjunto de dados e os rotulamos como pertencentes à classe $+1$. Os outliers foram colocados propositadamente longe da fronteira das classes encontrada para o caso livre de outliers; mais especificamente, na região de decisão da classe -1.

[2]Uma abordagem usual é fazer $\sigma = \text{MAR}/0.6745$, onde MAR é a mediana do resíduo absoluto.

Os resultados para o treinamento sem outliers são mostrados na Fig. 1a, na qual, como esperado, as curvas de decisão de ambos os classificadores são similares. Os resultados para o treinamento com outliers são mostrados em Fig. 1b, onde dessa vez a curva de decisão do classificador ELM padrão moveu (pendeu) em direção aos outliers, enquanto que a curva de decisão do classificador ROB-ELM permaneceu inalterada, revelando, portanto, a robustez a outliers da abordagem proposta. O conjunto de dados (com e sem outliers) usado nesse primeiro experimento pode ser disponibilizado pelos autores mediante solicitação.



(a) Conjunto de dados sem outliers.



(b) Conjunto de dados com outliers.

Figura 1. Curvas de decisão dos classificadores ELM padrão e ELM robusto proposto. (a) Conjunto de dados sem outliers. (b) Conjunto de dados com outliers.

Nos segundo e terceiro experimentos, visamos avaliar a robustez do classificador ROB-ELM usando um conjunto real de dados. Para esse experimento, quatro funções ponderadas (Bisquare, Fair, Huber e Logistic) foram testadas para implementação do classificador ROB-ELM, e a constante de regularização necessária à implementação do classificador ELM padrão foi setada para $\lambda = 10^{-2}$. O parâmetro de ajuste default k da função `robustfit` do Matlab foi adotado para todas as funções ponderadas.

A fim de avaliar a robustez a outliers do classificador, seguimos a metodologia introduzida por Kim e Ghahramani [9]. Assim, os rótulos originais de algumas amostras de dados de uma dada classe são deliberadamente alteradas para o rótulo da outra classe. Foi escolhido um conjunto de dados benchmarking (Ionosphere), que está publicamente disponível para download no site UCI Machine Learning Repository [3].

O conjunto de dados Ionosphere descreve uma tarefa de classificação binária, onde sinais de radar têm como alvo dois tipos de elétrons na ionosfera. Retorno de radar ”Bom”são aqueles que mostram evidência de algum tipo de estrutura na ionosfera. Retorno ”Ruim”são aqueles que não fazem isso, já que seus sinais passam através da ionosfera. Esse conjunto de dados é composto por 351 pontos de dados de dimensão 34, e duas classes (bom e ruim).

Rotulamos os pontos de dados da classe Bom ($N_g = 225$ amostras) e classe Ruim ($N_b = 126$ amostras) como $+1$ e -1, respectivamente. 80% das amostras disponíveis são selecionadas aleatoriamente para propósito de treinamento. Em adição, outliers são construídos selecionando, aleatoriamente, um certo percentual P_{out} das amostras de treinamento a partir da Classe $+1$, e alterando seus rótulos para a Classe -1. O conjunto de teste é livre de outliers, já que o objetivo do experimento é avaliar a influência de outliers na construção das fronteiras de decisão dos classificadores. Para esse propósito, avaliamos os desempenhos dos classificadores ELM e ROB-ELM para $P_{out} = 5\%, 10\%$ e 20%, e para valores diferentes de q (número de neurônios ocultos). Os resultados são apresentados nas Tabelas I, II e III. Nessas tabelas, mostramos os valores das taxas de classificação e desvios padrão correspondentes, a partir de 100 rodadas treinamento/teste.

Analisando os resultados, podemos verificar primeiramente que os desempenhos de todas as variantes do classificador proposto ROB-ELM tendem a melhorar com um aumento no número de neurônios ocultos. Segundo, os desempenhos deterioram com um aumento no número de outliers, como esperado.

Como um resultado importante, podemos facilmente verificar que os desempenhos do classificador ROB-ELM são melhores que o do ELM padrão, especialmente ao usar as funções ponderadas Fair, Huber e Logistic. Enquanto as melhorias nos desempenhos do classificador ROB-ELM são maiores para valores maiores de q (número de neurônios ocultos), não existem melhorias significativas no classificador ELM padrão quando q aumenta, especialmente para $P_{out} = 10\%$ and 20%.

Como uma observação final, vale a pena mencionar mais uma vez que os excelentes desempenhos do classificador ROB-ELM proposto foram obtidos usando os valores default do parâmetro de ajuste k, da função `robustfit` do Matlab, para todas as funções ponderadas usadas nesse artigo. Isso é particularmente interessante para o usuário prático que deseja

Tabela I
COMPARAÇÃO DE DESEMPENHO DOS CLASSIFICADORES ELM E ROB-ELM ($P_{out} = 5\%$).

q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	70.94±4.95	69.77±6.72	72.21±4.66	72.30±4.27	72.23±4.05
15	70.67±4.53	70.76±5.54	73.41±3.34	73.11±3.40	73.56±3.75
20	70.93±3.90	72.29±5.87	74.00±3.49	74.07±3.39	74.44±3.40
25	71.24±3.33	72.89±4.86	73.94±2.94	74.51±3.35	74.41±2.87
30	71.90±3.16	72.06±4.54	75.50±2.56	75.81±2.19	75.30±2.41
35	71.81±3.01	73.90±3.49	76.36±1.75	76.13±1.51	76.27±1.62
40	72.73±2.53	73.87±3.35	76.20±2.10	76.69±1.96	76.60±2.03
45	73.34±2.25	73.46±4.05	76.69±1.83	76.83±1.86	76.93±1.89
50	73.27±2.33	73.71±4.36	77.21±2.02	76.80±2.07	76.76±2.18
100	73.97±2.03	71.49±5.42	77.03±3.68	76.46±4.32	77.03±4.17

Tabela II
COMPARAÇÃO DE DESEMPENHO DOS CLASSIFICADORES ELM E ROB-ELM ($P_{out} = 10\%$).

q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	67.03±5.12	69.77±4.80	71.06±4.52	70.57±4.77	70.47±4.03
15	66.39±5.35	70.61±4.94	70.73±4.79	68.79±5.03	70.63±4.73
20	65.57±5.71	72.04±4.77	70.86±4.06	70.56±4.76	70.90±3.98
25	65.93±4.50	73.37±4.25	72.03±3.19	71.64±3.81	71.49±4.08
30	65.83±4.07	72.17±3.69	73.17±3.09	72.40±3.48	73.04±3.08
35	66.51±3.82	73.46±3.26	73.89±2.79	73.17±2.71	73.81±2.68
40	67.17±2.78	74.56±3.01	74.86±2.94	75.07±2.78	74.60±2.90
45	66.96±3.00	75.23±3.73	75.23±3.22	74.67±2.95	75.63±2.72
50	67.27±2.57	75.33±3.66	75.67±2.79	75.66±3.14	75.97±2.74
100	68.81±2.37	72.99±6.02	77.07±4.41	76.90±4.19	77.01±4.50

Tabela III
COMPARAÇÃO DE DESEMPENHO DOS CLASSIFICADORES ELM E ROB-ELM ($P_{out} = 20\%$).

q	ELM ($\lambda = 0.01$)	ROB-ELM (Bisquare)	ROB-ELM (Fair)	ROB-ELM (Huber)	ROB-ELM (Logistic)
10	50.03±5.19	50.93±5.22	55.16±6.95	50.51±5.82	53.20±6.93
15	50.07±4.49	51.43±5.53	55.20±4.86	50.79±4.64	53.87±4.99
20	48.63±3.63	52.39±4.99	55.99±6.00	50.51±4.33	54.74±5.05
25	49.83±4.32	52.31±4.25	56.46±5.64	50.90±4.41	54.11±4.49
30	49.49±3.35	53.21±3.47	56.93±4.38	51.26±4.00	55.26±3.94
35	49.90±3.22	54.76±4.39	58.71±4.90	52.84±4.37	56.69±4.62
40	49.24±2.79	57.53±5.80	61.11±6.28	56.37±5.44	59.03±6.09
45	49.93±3.03	58.41±6.83	62.76±6.92	58.69±6.73	61.04±5.87
50	50.16±3.02	61.93±6.68	64.51±7.83	60.76±7.40	63.61±6.82
100	51.43±3.18	68.94±7.89	69.91±6.97	67.03±9.39	68.39±7.67

obter resultados rápidos e precisos, sem gastar muito tempo em longas rodadas de ajuste fino do classificador.

V. CONCLUSÃO

No artigo introduzimos um classificador ELM (ROB-ELM) para classificação supervisionada de padrões na presença de erros de rotulação (outliers) nos dados. O classificador ROB-ELM foi projetado por meio de métodos de estimação-M, os quais são usados para calcular o operador matriz peso, em vez de usar a solução de mínimos quadrados ordinários. Por meio de simulações computacionais em conjuntos de dados sintéticos e reais, mostramos que o classificador resultante é mais robusto a outliers que o classificador ELM padrão.

Atualmente, estamos avaliando ainda mais o desempenho do ROB-ELM em outros conjuntos de dados de classificação binária, e também em problemas multiclasse. Os resultados obtidos até agora sugerem que esta é uma abordagem promissora.

VI. AGRADECIMENTOS

Os autores agradecem à CAPES pelo apoio financeiro através do projeto PROCAD - NF 2009.

REFERÊNCIAS

[1] Deng, W., Zheng, Q., Chen, L.: Regularized extreme learning machine. In: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09). pp. 389–395 (2009)
[2] Fox, J.: Applied Regression Analysis, Linear Models, and Related Methods. Sage Publications (1997)
[3] Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
[4] Horata, P., Chiewchanwattana, S., Sunat, K.: Robust extreme learning machine. Neurocomputing 102, 31–44 (2012)
[5] Huang, G.B., Wang, D.H., Lan, Y.: Extreme learning machines: a survey. International Journal of Machine Learning and Cybernetics 2, 107–122 (2011)
[6] Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70, 489–501 (2006)
[7] Huber, P.J.: Robust estimation of a location parameter. Annals of Mathematical Statistics 35(1), 73–101 (1964)
[8] Huber, P.J., Ronchetti, E.M.: Robust Statistics. john Wiley & Sons, LTD (2009)

[9] Kim, H.C., Ghahramani, Z.: Outlier robust gaussian process classification. In: Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition (SSPR)'08. pp. 896–905 (2008)
[10] Lee, C.C., Chiang, Y.C., Shih, C.Y., Tsai, C.L.: Noisy time series prediction using m-estimator based robust radial basis function neural networks with growing and pruning techniques. Expert Systems and Applications 36(3), 4717–4724 (2009)
[11] Lee, C.C., Chung, P.C., Tsai, J.R., Chang, C.I.: Robust radial basis function neural networks. IEEE Transactions on Systems, Man, and Cybernetics - Part B 29(6), 674–685 (1999)
[12] Li, D., Han, M., Wang, J.: Chaotic time series prediction based on a novel robust echo state network. IEEE Transactions on Neural Networks and Learning Systems 23(5), 787–799 (2012)
[13] Liu, N., Wang, H.: Ensemble based extreme learning machine. IEEE Signal Processing Letters 17(8), 754–757 (2010)
[14] Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: OP-ELM: Optimally pruned extreme learning machine. IEEE Transactions on Neural Networks 21(1), 158–162 (2010)
[15] Miche, Y., van Heeswijk, M., Bas, P., Simula, O., Lendasse, A.: TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization. Neurocomputing 74(16), 2413–2421 (2011)
[16] Mohammed, A., Minhas, R., Jonathan Wu, Q.M., Sid-Ahmed, M.A.: Human face recognition based on multidimensional PCA and extreme learning machine. Pattern Recognition 44(10–11), 2588–2597 (2011)
[17] Neumann, K., Steil, J.: Optimizing extreme learning machines via ridge regression and batch intrinsic plasticity. Neurocomputing 102, 23–30 (2013)
[18] Zong, W., Huang, G.B.: Face recognition based on extreme learning machine. Neurocomputing 74(16), 2541–2551 (2011)