O Algoritmo de Treinamento: Máquina de Aprendizado Extremo *(Extreme Learning Machine - ELM)*

Thiago Henrique Cupertino

SCE5809 - Redes Neurais

23 de Novembro de 2010

Conteúdo

- Introdução
  - Desvantagens do Back-Propagation
- Máquina de Aprendizado Extremo
  - ELM: Teoria
  - Algoritmo ELM: Características
  - Matriz Pseudo-Inversa
- Resultados de Simulações
- Extensões do ELM
  - ELM Podado
- Demais Referências

Introdução

- Tradicionalmente, todos os parâmetros de uma rede unidirecional têm que ser ajustados;
- Métodos baseados em gradiente descendente têm sido usados em vários algoritmos de treinamento (p. ex., algoritmo Back-Propagation);
- Tais métodos consomem grande tempo de treinamento devido ao ajuste iterativo dos parâmetros.

Desvantagens do Back-Propagation

- Quando a taxa de treinamento η é muito pequena, o algoritmo de treinamento converge muito lentamente. Caso contrário, quando η é muito grande, o algoritmo se torna instável e a rede diverge;
- É indesejável que o algoritmo pare em um mínimo local distante do mínimo global;

Desvantagens do Back-Propagation

- Redes Neurais podem ser super-treinadas com o algoritmo BP de maneira que a generalização fique prejudicada (*overfitting*);
- Aprendizado baseado em gradiente descendente pode consumir demasiado tempo de treinamento em muitas aplicações.

Máquina de Aprendizado Extremo

- Esse algoritmo contorna as desvantagens citadas anteriormente;
- Foi desenvolvido para redes com apenas duas camadas: a camada de entrada e a camada escondida;

G.-B. Huang, Q.-Y. Zhu e C.-K. Siew, Extreme Learning Machine: Theory and Applications, Neurocomputing 70, 489-501 (2006).

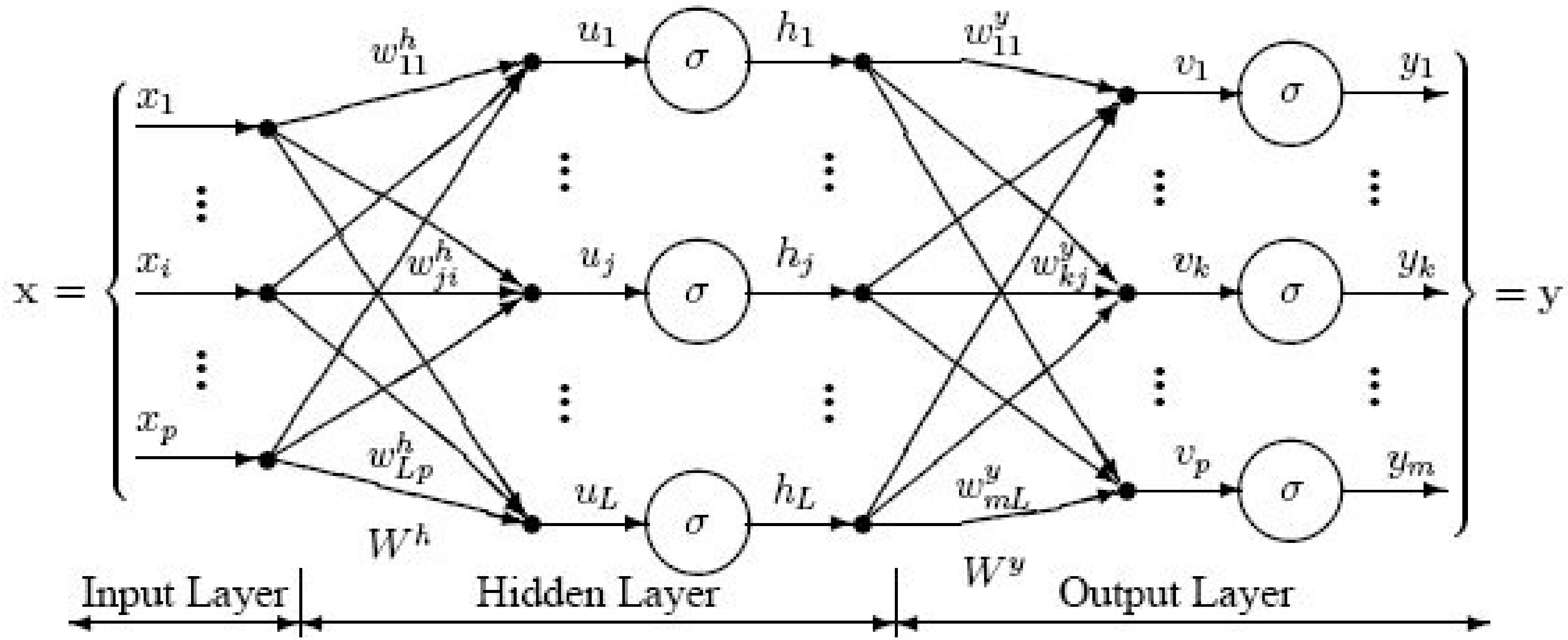
Máquina de Aprendizado Extremo

- Os pesos de entrada e os bias da camada escondida são escolhidos aleatoriamente;
- Os pesos da camada de saída são determinados ***analiticamente*** (i. e., não há ciclos iterativos para ajuste de parâmetros).

G.-B. Huang, Q.-Y. Zhu e C.-K. Siew, Extreme Learning Machine: Theory and Applications, Neurocomputing 70, 489-501 (2006).

ELM: Teoria

- Desenvolvido para redes com 2 camadas *(Single Layer Feedforward Network - SLFN)*:



ELM: Teoria

- Modelagem matemática:

$$\sum_{i=1}^{\tilde{N}} \beta_i g_i(\mathbf{x}_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{w}_i . \mathbf{x}_j + b_i) = \mathbf{o}_j, j = 1, \dots, N$$

$$\Downarrow$$

$$\sum_{i=1}^{\tilde{N}} \beta_i g(\mathbf{w}_i . \mathbf{x}_j + b_i) = \mathbf{t}_j, j = 1, \dots, N$$

- (x_j, t_j): *N* padrões de entrada;
- w_i: vetor peso do neurônio *i* da camada escondida;
- b_i: bias do neurônio *i* da camada escondida;
- Ñ: número de neurônios da camada escondida.
- β_i: vetor peso entre o neurônio escondido *i* e a camada de saída.

ELM: Teoria

- Em forma matricial:

$$\mathbf{H}\beta = \mathbf{T}$$

$$\mathbf{H}(\mathbf{w}_1,\ldots,\mathbf{w}_{\tilde{N}},b_1,\ldots,b_{\tilde{N}},\mathbf{x}_1,\ldots,\mathbf{x}_{\tilde{N}}) = \begin{bmatrix} g(\mathbf{w}_1.\mathbf{x}_1+b_1) & \cdots & g(\mathbf{w}_{\tilde{N}}.\mathbf{x}_1+b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ g(\mathbf{w}_1.\mathbf{x}_N+b_1) & \cdots & g(\mathbf{w}_{\tilde{N}}.\mathbf{x}_N+b_{\tilde{N}}) \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_1{}^T \\ \vdots \\ \beta_{\tilde{N}}{}^T \end{bmatrix}_{\tilde{N}xm}, \mathbf{T} = \begin{bmatrix} \mathbf{t}_1{}^T \\ \vdots \\ \mathbf{t}_{\tilde{N}}{}^T \end{bmatrix}_{Nxm}$$

ELM: Teoria

Teorema 2.1. *Dada uma SLFN com N neurônios na camada escondida e função de ativação $g : R \to R$ infinitamente diferenciável em qualquer intervalo, para N exemplos de treinamento distintos $(\mathbf{x}_i, \mathbf{t}_i)$, $\mathbf{x}_i \in R^n$ e $\mathbf{t}_i \in R^m$, para quaisquer $\mathbf{w}_i$ e b_i aleatoriamente selecionados dentro de quaisquer intervalos R^n e R, respectivamente, por qualquer função de distribuição de probabilidade, então com probabilidade 1, a matrix de saída da camada escondida H da SLFN é inversível e* $||\mathbf{H}\beta - \mathbf{T}|| = 0.$

S. Tamura e M. Tateishi, Capabilities of a Four-Layered Feedforward Neural Network: Four Layers Versus Three, IEEE Trans. Neural Networks 8 (2), 251-255 (1997).

G.-B. Huang, Learning Capability and Storage Capacity of Two-Hidden-Layer Feedforward Networks, IEEE Trans. Neural Networks 14 (2), 274-281 (2003).

ELM: Teoria

- Se o número de neurônios $\tilde{N}$ da camada escondida é igual ao número N de exemplos de treinamento, $N = \tilde{N}$, então a matriz H é quadrada e inversível quando o vetor de pesos w_i e os bias b_i são aleatoriamente escolhidos e, assim, as a SLFNs podem aprender estes exemplos de treinamento com erro zero.

ELM: Teoria

- Entretanto, na maioria dos casos o número de neurônios da camada escondida é muito menor do que o número de exemplos distintos de treinamento, $\tilde{N} << N$, e a matrix H não é quadrada;
- Solução de mínimos quadrados com a menor norma: $\beta = H^{\dagger}T$;
- $H^{\dagger}$: matriz inversa generalizada de Moore-Penrose da matriz H (pseudo inversa).

C. R. Rao e S. K. Mitra, Generalized Inverse of Matrices and its Applications, Wiley, New York (1971).

Algoritmo ELM

- INÍCIO
- Passo 1: Selecionar aleatoriamente valores para os pesos w_i e os bias b_i, $i = 1, ..., N$;
- Passo 2: Calcular a matriz de saída H da camada escondida.
- Passo 3: Calcular os pesos de saída $\beta = H^{\dagger}T$.
- FIM

Algoritmo ELM: Características

- *Menor erro de treinamento:* A solução $\beta = H^{\dagger}T$ é uma das soluções de mínimos quadrados de um sistema linear geral $H\beta = T$, o que significa que o menor erro de treinamento pode ser encontrado por esta solução.
- *Menor norma dos pesos:* Além disso, a solução $\beta = H^{\dagger}T$ tem a menor norma entre todas as soluções de mínimos quadrados de $H\beta = T$.
- A solução de menor norma é única.

P. L. Bartlett, The Sample Complexity of Pattern Classification with Neural Networks: The Size of the Weights is More Important than the Size of the Network, IEEE Trans. Inf. Theory 44 (2), 525-537 (2003).

Algoritmo ELM: Matriz Pseudo Inversa

- $H^{\dagger}$ satisfaz as seguintes propriedades:
- 1. $H\,H^{\dagger}\,H = H$
- 2. $H^{\dagger}\,H\,H^{\dagger} = H^{\dagger}$
- 3. $(H\,H^{\dagger})^{T} = H\,H^{\dagger}$
- 4. $(H^{\dagger}\,H)^{T} = H^{\dagger}\,H$
- Pode ser calculada eficientemente pelo método da Decomposição por Valores Singulares (*Singular Value Decomposition - SVD*).

Resultados de Simulações: sinc(x)

- Função sinc(x):

$$y(x) = \begin{cases} seno(x)/x, & x \neq 0 \\ 1, & x = 0. \end{cases}$$

Algorithms	Time (s)	
	Training	Testing
ELM	0.125	0.031
BP	21.26	0.032
SVR	1273.4	5.9087

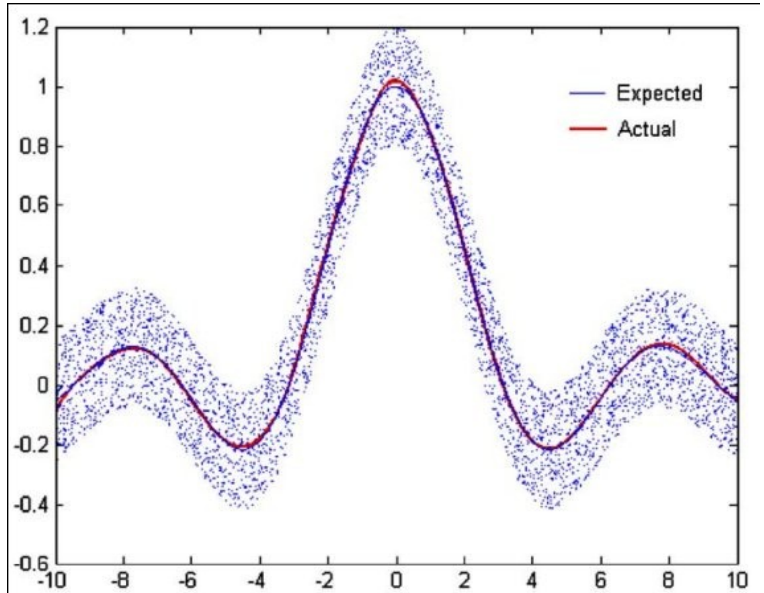
- ELM: 170 vezes mais rápido do que BP e 10000, do que SVR.

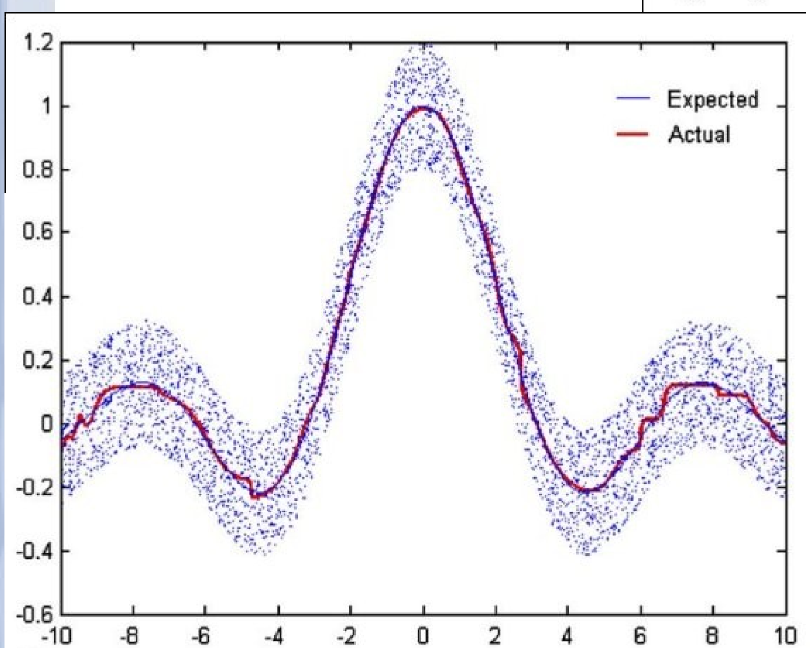
Resultados de Simulações: sinc(x)

Algorithms	Training	
	RMS	Dev
ELM	0.1148	0.0037
BP	0.1196	0.0042
SVR	0.1149	0.0007

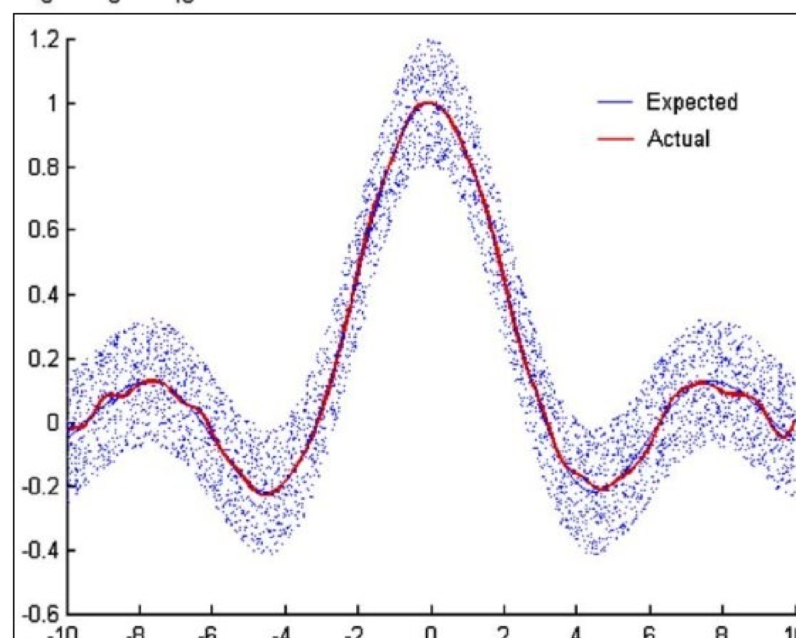
Algorithms	Testing		No of SVs/nodes
	RMS	Dev	
ELM	0.0097	0.0028	20
BP	0.0159	0.0041	20
SVR	0.0130	0.0012	2499.9

Resultados de Simulações: sinc(x)





(a) BP



(b) SVR

Resultados de Simulações: Vegetações Florestais

- Compreende em 581.012 instâncias e 54 atributos para cada instância. Classificação consistiu em separar a classe 2 das demais 6 classes.

Algorithms	Time (min)	
	Training	Testing
ELM	1.6148	0.7195
SLFN	12	N/A
SVM	693.6000	347.7833

- Tempo ELM 430 vezes menor do que SVM.

R. Collobert, S. Bengio e Y. Bengio, A Parallel Mixtures of SVMs for Very Large Scale Problems, Neural Comput. 14, 1105-1114 (2002).

Resultados de Simulações: Vegetações Florestais

Algorithms	Success rate (%)	
	Training	
	Rate	Dev
ELM	92.35	0.026
SLFN	82.44	N/A
SVM	91.70	N/A

Algorithms			# SVs/nodes
	Testing		
	Rate	Dev	
ELM	90.21	0.024	200
SLFN	81.85	N/A	100
SVM	89.90	N/A	31,806

- Generalização do ELM é melhor.

Extensões do ELM

- ELM com Critérios de Poda: melhora casos de overfitting, underfitting e generalização;
- ELM Evolutivo: melhora generalização e diminui tamaho da rede;
- ELM Baseado em Método de Otimização: estudo teórico comparando ELM e SVM.

Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten e A. Lendasse, OP-ELM: Optimally Pruned Extreme Learning Machine, IEEE Trans. Neural Networks 21 (1), 158-162 (2010).

H. J. Rong, Y.-S. Ong, A.-W. Tan e Z. Zhu, A Fast Pruned-Extreme Learning Machine for Classification Problem, Neurocomputing 72 (1-3), 359-366 (2008).

Q.-Y. Zhu, A. K. Qin, P. N. Suganthan e G.-B. Huang, Evolutionary Extreme Learning Machine, Pattern Recognition 38, 1759-1763 (2005).

G.-B. Huang, X. Ding e H. Zhou, Optimization Method Based Extreme Learning Machine for Classification, Neurocomputing (Article in Press), (2010).

Optimally Pruned ELM

- Erro Médio Quadrático (negrito) e Desvio Padrão

	Abalone	Ailerons	Elevators	Computer	Auto P.	CPU	Servo	Breast C.	Bank	Stocks	Boston
SVM	**4.5**	**1.3e-7**	**6.2e-6**	**1.2e+2**	**2.8e+7**	**6.5e+3**	**6.9e-1**	**1.2e+3**	**2.7e-2**	**5.1e-1**	**3.4e+1**
	2.7e-1	2.6e-8	6.8e-7	8.1e+1	8.4e+7	5.1e+3	3.3e-1	7.2e-1	8.0e-4	9.0e-2	3.1e+1
OPELM	**4.9**	**2.8e-7**	**2.0e-6**	**3.1e+1**	**9.5e+7**	**5.3e+3**	**8.0e-1**	**1.4e+3**	**1.1e-3**	**9.8e-1**	**1.9e+1**
	6.6e-1	1.5e-9	5.4e-8	7.4	4.0e+6	5.2e+3	3.3e-1	3.6e+2	1.0e-6	1.1e-1	2.9
ELM	**8.3**	**3.3e-8**	**2.2e-6**	**4.9e+2**	**7.9e+9**	**4.7e+4**	**7.1**	**7.7e+3**	**6.7e-3**	**3.4e+1**	**1.2e+2**
	7.5e-1	2.5e-9	7.0e-8	6.2e+1	7.2e+9	2.5e+4	5.5	2.0e+3	7e-4	9.35	2.1e+1
GP	**4.5**	**2.7e-8**	**2.0e-6**	**7.7**	**2.0e+7**	**6.7e+3**	**4.8e-1**	**1.3e+3**	**8.7e-4**	**4.4e-1**	**1.1e+1**
	2.4e-1	1.9e-9	5.0e-8	2.9e-1	1.0e+7	6.6e+3	3.5e-1	1.9e+2	5.1e-5	5.0e-2	3.5
MLP	**4.6**	**2.7e-7**	**2.6e-6**	**9.8**	**2.2e+7**	**1.4e+4**	**2.2e-1**	**1.5e+3**	**9.1e-4**	**8.8e-1**	**2.2e+1**
	5.8e-1	4.4e-9	9.0e-8	1.1	9.8e+6	1.8e+4	8.1e-2	4.4e+2	4.2e-5	2.1e-1	8.8

Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten e A. Lendasse, OP-ELM: Optimally Pruned Extreme Learning Machine, IEEE Trans. Neural Networks 21 (1), 158-162 (2010).

Outras Referências

- Artigos, conferências e códigos-fonte:
- http://www3.ntu.edu.sg/home/egbhuang/

FIM