Universidade de São Paulo – São Carlos / SP
Instituto de Ciências Matemáticas e de Computação

Echo State Networks

Sistemas Inteligentes

Prof. Dra. Roseli A. Francelin Romero

Fabrício Breve
João R. Bertini Jr.

Sumário

Echo State Networks

Resumo

O modelo de rede neural Echo State Network (ESN) foi recentemente proposto como um modelo alternativo de rede neural recorrente. O modelo é motivado pela ineficácia dos algoritmos atualmente existentes para treinamento de redes neurais recorrentes. Uma ESN é composta, basicamente, por uma camada de entrada, um reservatório de neurônios arranjados de maneira recorrente, e uma camada de saída. Como somente os pesos da camada de saída são atualizados durante o treinamento, a tarefa de aprendizado da rede torna-se linear. A principal vantagem do modelo ESN é a habilidade de modelar sistemas sem a necessidade treinar os pesos recorrentes.

1. Introdução

Echo State Networks, proposto em [Jaeger 2001], é um modelo de rede neural artificial recorrente (RNR). RNRs caracterizam-se pela presença de ao menos um laço de realimentação ("recorrência") em seus neurônios. Como descrito em [Haykin 1999], RNRs podem ser definidas como rede neurais que usam saídas de ao menos um neurônio da rede no tempo t, como entrada para outros neurônios no tempo t + 1. RNRs podem manter a ativação mesmo na ausência da entrada e assim exibir memória dinâmica. As redes neurais biológicas são tipicamente recorrentes. Como redes neurais biológicas, uma RNR artificial pode aprender a imitar um sistema alvo.

As RNRs geralmente apresentam um comportamento dinâmico não-linear, o que as torna mecanismos úteis na predição de valores inerentemente temporais (e.g. séries temporais). O atraso no fornecimento de entradas à rede neural introduz memória na rede, proporcionando aos neurônios valores de entrada atuais e valores temporalmente anteriores a eles. Diversos algoritmos de aprendizagem conhecidos (ver [Willian & Zipser 1989], [Werbos 1990], por exemplo) adaptam de maneira incremental os pesos sinápticos de um RNR. Estes algoritmos, no entanto, têm uso limitado devido a problemas do tipo: 1) convergência para uma solução é lenta; 2) as soluções encontradas, muitas vezes são soluções sub-ótimas e 3) a

maioria dos modelos recorrentes requerem convergência a pontos estáveis [Hertz *et al.* 1991], fato que limita a capacidade de aprendizado da rede.

Uma das principais diferenças entre a abordagem ESN e outros métodos de RNR estão no fato da primeira permitir o uso de grande número de neurônios (da ordem de 50 a 1000 neurônios, técnicas precedentes usam tipicamente 5 a 30 neurônios) e dessa forma proporcionar maior dinâmica à rede. Outra diferença significante em relação a outros modelos de redes neurais recorrentes é que em uma rede ESN somente as conexões do reservatório para a saída são treinadas. Fato este responsável por tornar este modelo de rede atrativo, uma vez que a ausência de dependência cíclica entre as conexões treinadas da camada de saída (readout) permite que o treinamento transforme-se em uma tarefa simples da regressão linear.

No que segue será apresentado, de forma simplista, o modelo de rede neural ESN seguido por um exemplo pratico retirado de [Jaeger 2002]. A teoria completa sobre ESN, referencia e exemplos podem ser encontrados em [Jaeger 2001] e [Jaeger 2002]. Um modelo similar ao de ESN com embasamento mais biológico foi recentemente proposto em [Maass *et al.* 2002] [Maass *et al.* 2002a] e é chamado de redes de estado liquido (liquid state networks).

2. A Propriedade de Echo State

A propriedade de echo state é crucial para que o modelo ESN funcione. Intuitivamente, uma RNR exposta a sinais externos u(n) tem a propriedade de echo state se as ativações dos neurônios x(n) forem variações sistemáticas dos sinais de entrada u(n). De uma maneira mais formal, isso significa que para cada neurônio interno x_i existe uma função de eco (echo function) e_i, tal que, se a rede foi executada por um tempo indefinidamente longo no passado, o estado corrente poderá ser escrito como na equação (1).

$$x_i(n) = e_i(u(n), u(n-1), u(n-2), \ldots) \qquad (1)$$

Para ESN de tempo discreto existem diversas definições alternativas não-triviais e caracterizações algébricas de que as matrizes de pesos W da rede conduzem às redes que têm a propriedade de echo state [Jaeger 2001]. Para finalidades práticas, entretanto, basta ajustar o raio espectral $\rho(W)$ de W a um valor abaixo da unidade para garantir a propriedade de echo state. É também importante que a dinâmica dos neurônios do reservatório seja bastante

variada. Para isso é necessário que os neurônios do reservatório apresentem um padrão de interconexão esparsa (cerca de 1-20%).

3. Estrutura de uma rede ESN

Considere uma rede neural de tempo discreto com K entradas, N neurônios ocultos e L neurônios na camada de saída. Como mencionado anteriormente, as entradas no tempo n são representadas por $u(n) = (u_1(n),..., u_K(n))$ e o estado de ativação dos neurônios ocultos (pertencentes ao reservatório) por $x(n) = (x_1(n), ..., x_N(n))$ e dos neurônios de saída $y(n) = (y_1(n), ..., y_L(n))$.

As conexões são dadas por pesos reais e podem ser representadas de forma matricial. Os pesos da entrada para o reservatório são armazenados em uma matriz de pesos N x K, notada por W_{in}. Os pesos do reservatório constituem uma matriz N x N, notados por W, e os pesos do reservatório para os neurônios de saída constituem a matriz L x (K + N + L) chamada W_{out}.

Note, na Figura 1 que algumas conexões estão marcadas em pontilhado, essas conexões são opcionais no modelo original. As conexões da entrada, diretamente para a saída, quando existente, são armazenadas em W_{out} e as conexões da saída para o reservatório bem como as conexões entre os neurônios de saída constituem a matriz W_{back}. Os pesos em toda a rede possuem valores reais, e são gerados aleatoriamente no inicio do treinamento. Os únicos pesos que são treinados são os pesos do reservatório para a saída (W_{out}), os pesos do reservatório W, devem ser definidos como mencionado na seção anterior e os pesos de W_{in} são obtidos, na maioria das vezes de maneira aleatória. A Figura 1 resume a arquitetura de uma rede ESN considerada aqui.

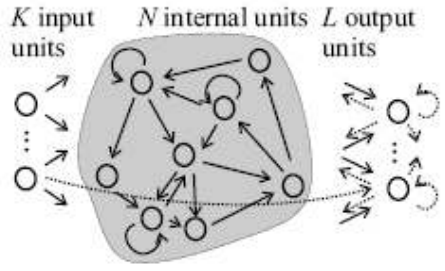

Figura 1. Esquema geral de uma rede ESN

O estado de ativação definido para os neurônios pertencentes ao reservatório é ilustrado na equação (2).

$$x(n+1) = f(W_{in}u(n+1) + Wx(n) + W_{back}y(n)) \quad (2)$$

Na qual x(n+1) representa o estado de ativação de um neurônio do reservatório no tempo n+1; a função f é a função tangente hiperbólica (embora em principio, qualquer função diferenciável possa ser utilizada). As matrizes W_{in} W e W_{back} correspondem, aos pesos de entrada, pesos do reservatório e pesos da saída para o reservatório[1], respectivamente.

Já o estado de ativação dos neurônios de saída no tempo n + 1, y(n+1) é definido de acordo com a equação (3), o símbolo '|' representa, neste caso, concatenação de matrizes.

$$y(n+1) = f_{out}(W_{out}(u(n+1) | x(n+1) | y(n))) \quad (3)$$

Na equação (3), a função f_{out} é, também a função tangente hiperbólica e a matriz W_{out} é a matriz de pesos dos neurônios do reservatório para a saída, esta é a única matriz modificada durante o treinamento.

4. Treinamento das ESN

Em [Jaeger 2002] é apresentado informalmente os princípios das ESN mostrando como treinar uma rede neural para gerar uma onda senoidal:

A onda desejada é dada por $d(n) = \frac{1}{2}\sin\left(\frac{n}{4}\right)$. Esta tarefa não envolve entrada, portanto teremos uma rede sem nenhuma entrada, e com uma única saída que após o treinamento produzirá $d(n)$. O sinal "professor" é uma seqüência de 300 passos de $d(n)$.

Inicialmente é construída uma rede recorrente com 20 neurônios, cujos pesos das conexões internas W são inicializados com valores aleatórios e não muda durante o treinamento. Esta rede é chamada de "reservatório dinâmico". Os neurônios são sigmoidais com função de ativação f = tanh.

Essa construção usando valores aleatórios para W poderia fazer com que a rede apresentasse um comportamento oscilatório ou mesmo caótico, e isso não é o desejado. A solução é usar valores baixos para W, quanto mais baixos esses pesos, mais a rede tende a

[1] Os peso da saída para o reservatório (W_{back}) são opcionais.

converger para um estado zero iniciando a partir de qualquer estado arbitrário. Assumimos então uma inicialização de W com valores baixos, a Figura 2 mostra o gráfico da saída dos 20 neurônios quando iniciados com um valor aleatório x(0).

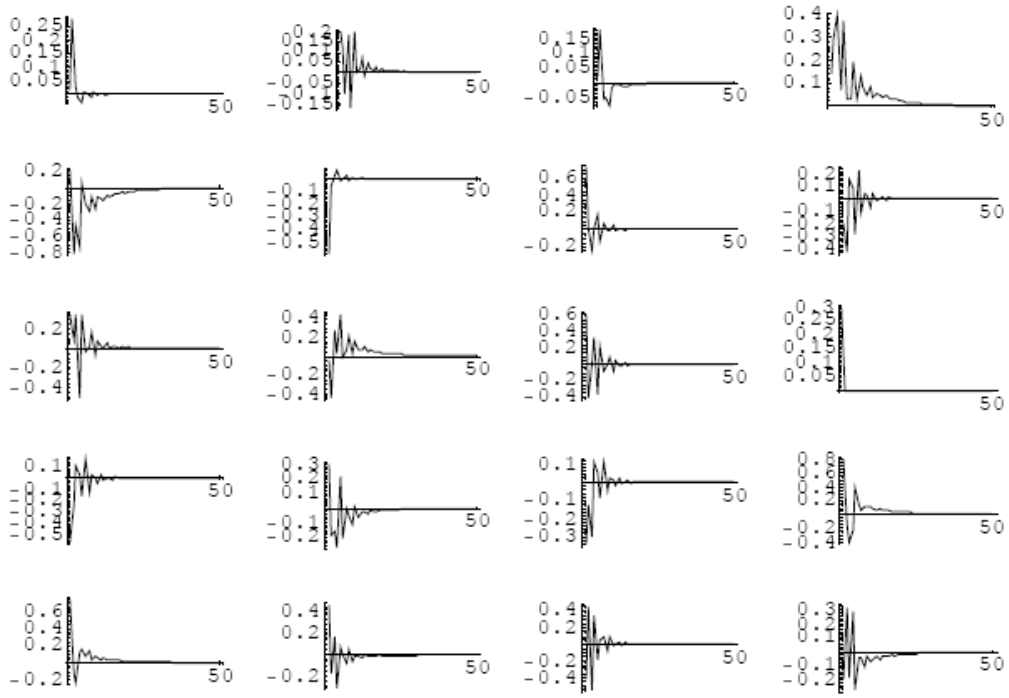

Figura 2. Dinâmica dos neurônios no reservatório dinâmico.

Adicionamos um único neurônio de saída para este reservatório. Esta saída tem conexões que projetam de volta para o reservatório. A estas retroprojeções são atribuídos pesos aleatórios W_{back}, que também são fixos e não mudam durante o treinamento. O neurônio de saída é linear.

As únicas conexões que são modificadas durante o aprendizado são as do reservatório para o neurônio de saída, cujos pesos são dados por W_{out}. Esses pesos não são definidos nem usados durante o treinamento. A Figura 3 mostra a rede preparada para o treinamento.

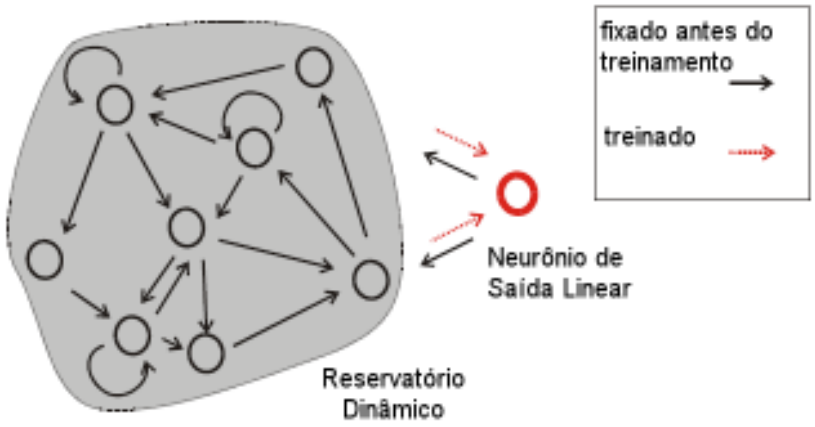

Figura 3. Configuração da ESN para treinar o gerador de ondas senoidais

O treinamento é feito em duas etapas: *amostragem* e *computação de pesos*. Veremos estas duas etapas a seguir.

4.1 Amostragem

Durante a fase de amostragem, o sinal professor é na unidade de saída para os tempos $n = 1,...,300$. A rede é iniciada no passo $n = 1$ com um estado arbitrário (aqui usaremos o estado zero). O sinal professor $d(n)$ é bombeado no reservatório através das conexões de retroprojeção W_{back} e, portanto estimula uma dinâmica de ativação dentro do reservatório. A Figura 4 mostra o que acontece dentro do reservatório nos passos $n = 100,...,150$.

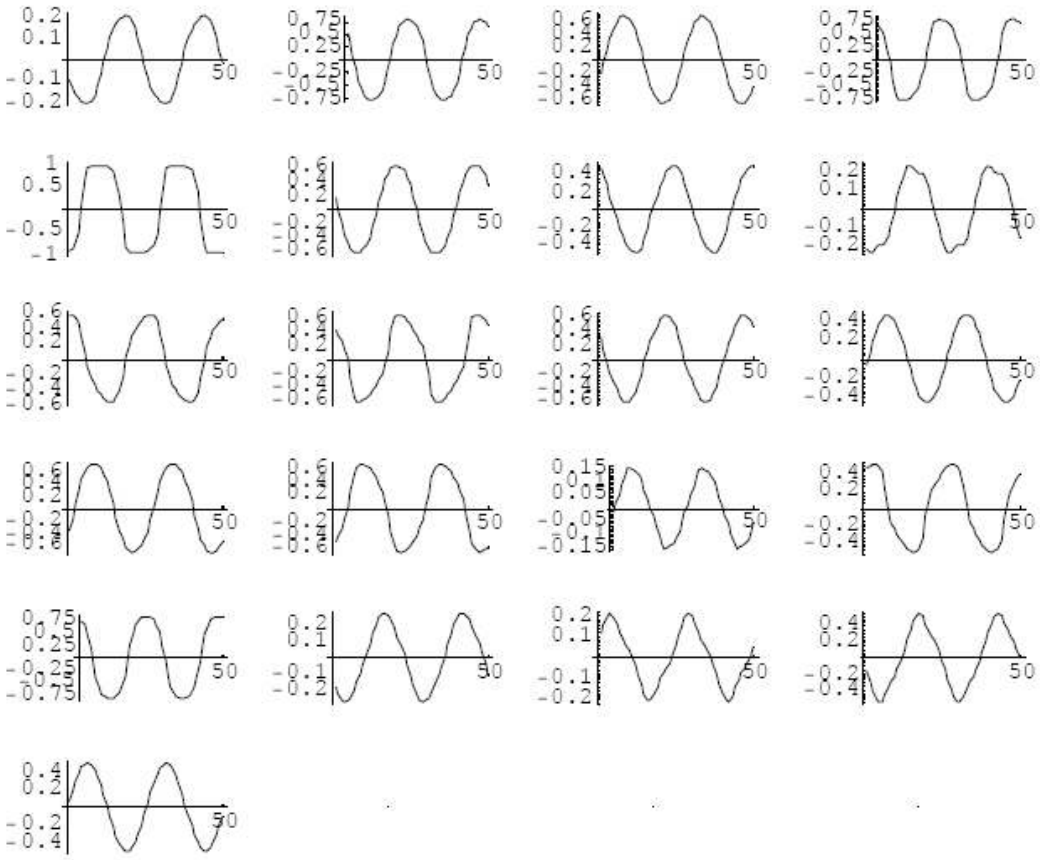

Figura 4. As saídas de cada um dos 20 neurônios do reservatório induzidas por um sinal professor $d(n)$ no neurônio de saída (último gráfico).

Neste ponto é importante observar que os padrões de ativação dentro do reservatório são periódicos, cujos períodos têm a mesma freqüência do sinal professor, e que estes padrões diferem um dos outros dentro do reservatório.

Durante o período de amostragem, os sinais internos $x(n) = (x_1(n),...,x_{20}(n))$ para $n = 101,...,300$ são coletados nas linhas de uma matriz M de tamanho 200x20. Ao mesmo tempo, as saídas do professor $d(n)$ são coletadas nas linhas de uma matriz T de tamanho 200x1.

Não são coletados dados de $n = 1,...,100$, porque nesses passos iniciais a dinâmica da rede é parcialmente determinada pelos estados iniciais arbitrários. No passo n = 100 é seguro afirmar que os efeitos da inicialização arbitrária já desapareceram e que os estados da rede são determinados apenas por $d(n)$ como mostrado na Figura 3.

4.2 Computação de pesos

Agora chegou a hora de computar os 20 pesos de saída w_i^{out} para nosso neurônio linear de saída $y(n)$ tais que a saída do professor $d(n)$ seja aproximada como uma combinação linear das séries de ativação internas $x_i(n)$, de acordo com a equação (4).

$$d(n) \approx y(n) = \sum_{i=1}^{20} w_i^{out} x_i(n) \tag{4}$$

De maneira mais específica, computamos os pesos w_i^{out} tais que o erro de treinamento quadrático médio é minimizado, como mostra a equação (5).

$$\text{MSE}_{train} = \frac{1}{200} \sum_{n=101}^{300} (d(n) - y(n))^2 = \frac{1}{200} \sum_{n=101}^{300} \left(d(n) - \sum_{i=1}^{20} w_i^{out} x_i(n) \right)^2 \tag{5}$$

Do ponto de vista matemático esta é uma tarefa de regressão linear, que consiste em computar os pesos de regressão w_i^{out} para uma regressão de $d(n)$ nos estados da rede $x_i(n)$ com $n = 101,...,300$

De um ponto de vista intuitivo-geométrico, isto significa combinar os 20 estados internos vistos na Figura 3 de forma que a combinação resultante melhor aproxime o sinal do professor visto na mesma figura (o último).

E finalmente do ponto de vista algorítmico, a computação *offline* dos pesos de regressão equivalem a computar uma pseudo-inversa: os pesos desejados que minimizam MSE_{train} são obtidos multiplicando a pseudo-inversa de M com T (equação 6).

$$\text{W}_{out} = \text{M}^{-1}\text{T} \tag{6}$$

Computar a pseudo-inversa de uma matriz é uma operação padrão na álgebra linear. Neste exemplo, o erro de treinamento com os pesos ótimo foi $\text{MSE}_{train} = 1,2^{-13}$. Uma vez calculados os pesos, estes são inclusos na rede, a qual estará pronta para uso.

4.3 Utilização

Após os pesos de saída serem escritos nas conexões de saída, a rede foi executada por mais 50 passos, continuando do último estado de treinamento $\mathrm{x}(300)$, porém agora a saída do professor não será mais utilizada. Dessa forma a saída da rede $y(n)$ passa a ser gerada pela própria rede treinada. O erro de teste é dado pela equação (7).

$$\mathrm{MSE}_{test} = \frac{1}{50}\sum_{n=301}^{350}(d(n)-y(n))^2 \tag{7}$$

Da equação (7) foi encontrado $\mathrm{MSE}_{test} = 5{,}6^{-12}$, que é maior que o erro de treinamento, mas ainda bastante pequeno, mostrando que a rede aprendeu a gerar a onda senoidal de forma precisa.

É possível afirmar, portanto, que o sinal $y(n)$ é obtido através de seus próprios ecos $x_i(n)$.

Para mostrar que a estabilidade obtida não está no fato de os passos avaliados começarem em $n = 301$, que é gerado pela saída do professor [Jaeger 2002] inicializa a rede treinada a partir de um estado arbitrário, e mesmo assim após alguns passos ela se estabiliza no sinal desejado, conforme mostrado na Figura 5.

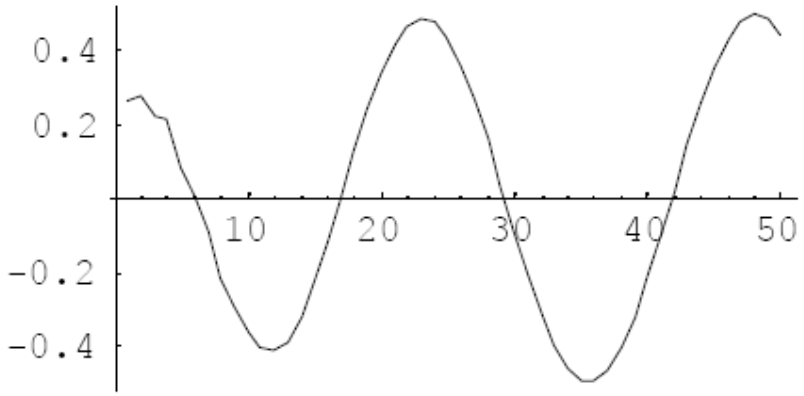

Figura 5. Iniciando a rede treinada a partir de um estado aleatório. O gráfico mostra as primeiras 50 saídas.

Referências

[Haykin 1999] Haykin, S. Neural Networks: A Comprehensive Foundation, Prentice Hall, 2nd Edition, 1999.

[Hertz et al. 1991] Hertz, J., Krogh, A., Palmer, R. Introduction to the theory of neural computation, Addison-Wesley (1991)

[Jaeger 2001] Jaeger, H. The "echo state" approach to analyzing and training recurrent neural networks. GMD Report 148, GMD – German National Research Institute for Computer Science, [http://www.gmd.de/People/Herbert.Jaeger/Publications.html], 2001.

[Jaeger 2002] Jaeger, H. Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKT and the echo state networks approach (revised version). GMD Report 159, Fraunhofer Institute AIS, 2002.

[Jaeger 2003] Jaeger, H. Adaptative nonlinear system identification with echo state networks, In: *Advances in Neural Information Processing Systems*, MIT-Press, Cambridge, M.A., pp. 593-600, 2003.

[Maass *et al.* 2002] Maass, W., Natschlaeger, T., Markram, H. Real-time computing without stable states: A new framework for neural computation based on permutations. [http://www.cis.tugraz.at/igi/maass/psfiles/LMS-v106.pdf], 2002.

[Maass *et al.* 2002a] Maass, W., Natschlaeger, T., Markram, H. A model for real-time computing in generic neural microcircuits. *Advances in Neural Information Processing System* 15 (Proc. NIPS). MIT Press, 2002.

[Werbos 1990] Werbos, P. J. Back propagation through time: What it does and how to do it, Proc. IEEE, vol. 78 No. 10, pp. 1550–1560, 1990.

[Willian & Zipser 1989] Williams, R. J., Zipser, D. A learning algorithm for continually running fully recurrent neural networks, *Neural Computation*, vol. 1, pp. 270-280, 1989