Deep Learning Technology and Application

Ge Li

Peking University

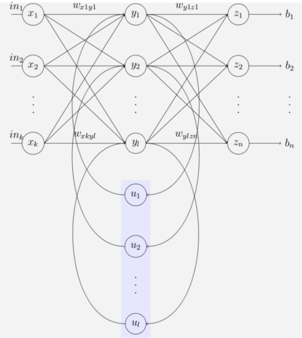
Table of contents

Is Feed-forward Neural Networks Powerful Enough?

- Feed-forward Neural Networks isn't Powerful Enough
  - There were no undirected cycles in the connectivity patterns;
  - Although neurons in the brain do contain undirected cycles as well as connections within layers;
  - We chose to impose these restrictions to simplify the training process at the expense of computational versatility.
- In order to create more powerful compuational systems, **directed cycles should be allowed in neural networks**
  - neurons can be connected to themselves.
  - RNNs are this kind of NN

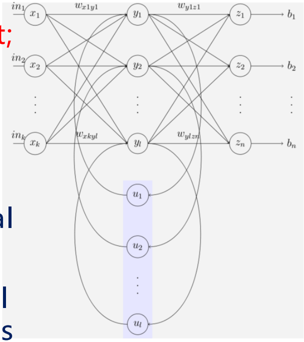
Is Feed-forward Neural Networks Powerful Enough?

- To create more powerful compuational systems, we allow RNNs to break these artificially imposed rules.
  - Thus, RNNs do not have to be organized in layers and directed cycles are allowed.
  - In fact, **neurons are actually allowed to be connected to themselves**.
  - One quite promising solution to tackling the problem of learning sequences of information.

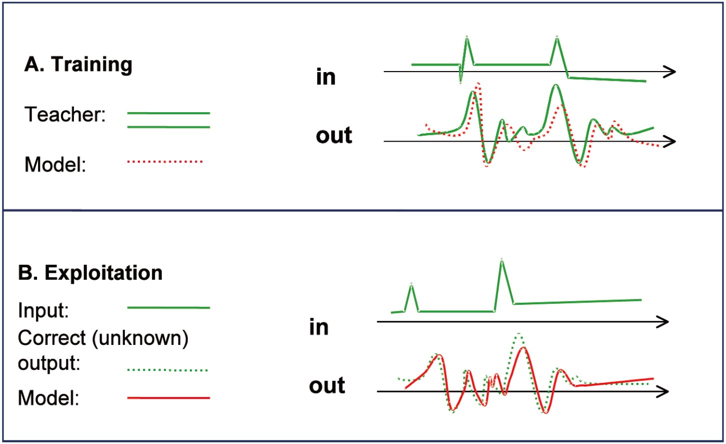


Recurrent Neural Network

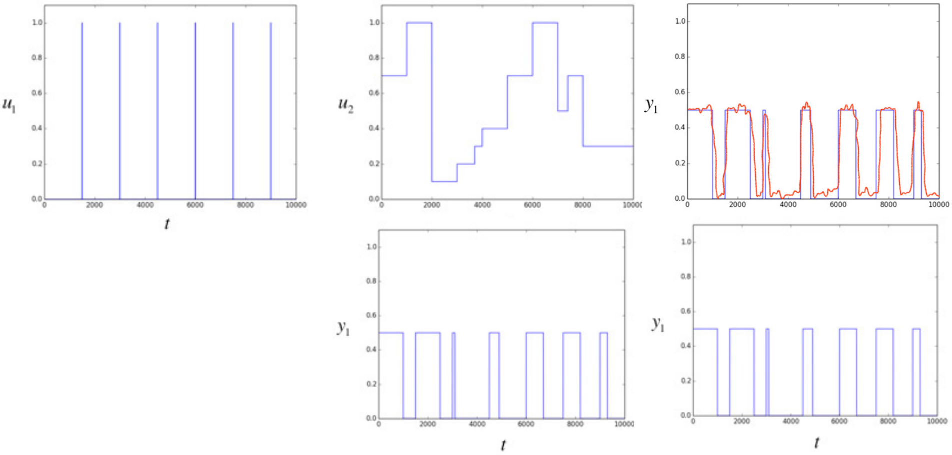
- RNN(Recurrent Neural Network)
  - all biological neural networks are recurrent;
  - RNNs implement dynamical systems mathematically, ;
- Some Old View :
  - RNNs can approximate **arbitrary** dynamical systems with **arbitrary** precision;
  - RNNs are (not often) proposed in technical articles as "in principle promising" solutions for difficult tasks.



RNN的作用

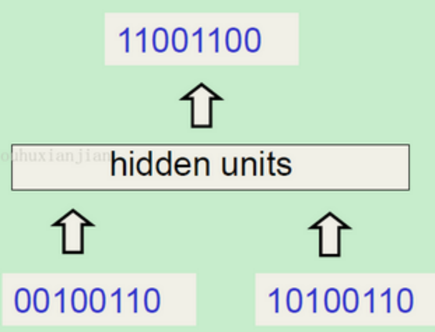

A. Training
Teacher:
Model:
in
out
B. Exploitation
Input:
Correct (unknown)
output:
Model:
in
out

一个典型的例子



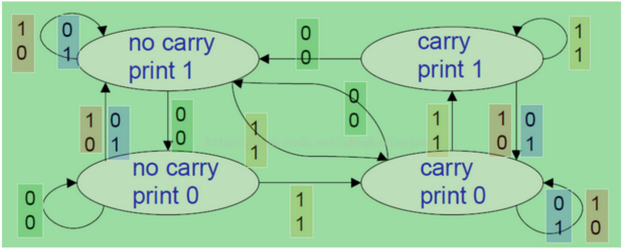
一个应用示例

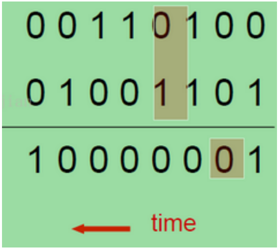
- 想一下
  - 如何构造一个前馈神经网络完成如下的学习？有什么问题？



一个应用示例

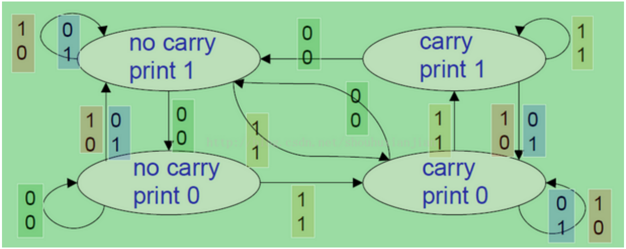
- 想一下
  - 如何构造一个前馈神经网络完成如下的学习？有什么问题？

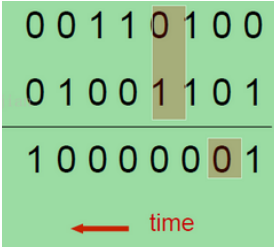




一个应用示例

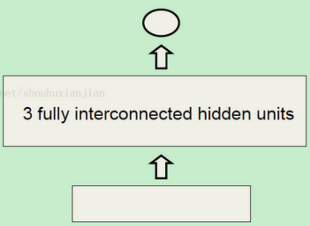
- 是否可以用RNN完成？
  - 几个输入？
  - 几个隐藏层？

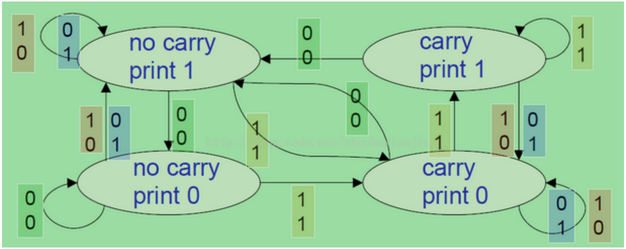


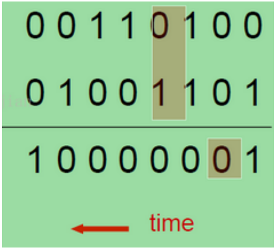


一个应用示例

- 是否可以用RNN完成？
  - 几个输入？
  - 几个隐藏层？

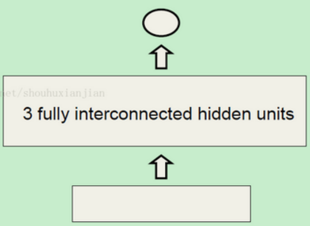


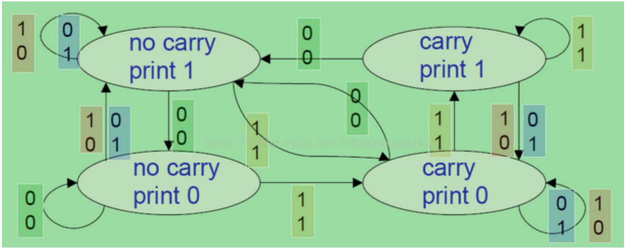


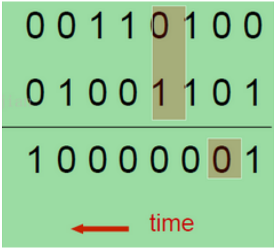


一个应用示例

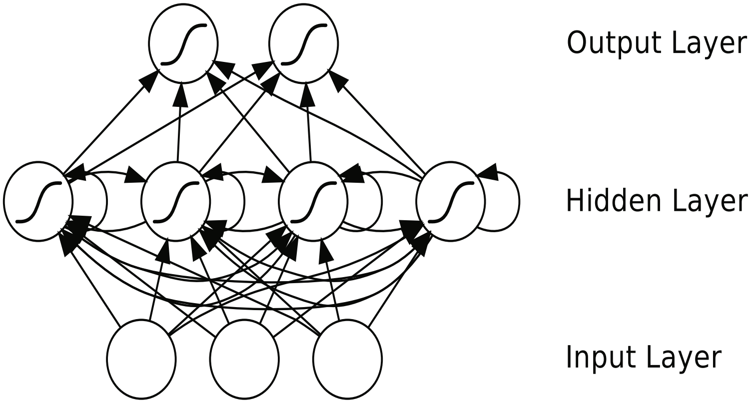
- RNN的使用场景
  - 需要抓取时间序列中隐藏的规律
  - 拓展到其他线性序列规律
  - “规律”隐藏在神经网络的循环层中



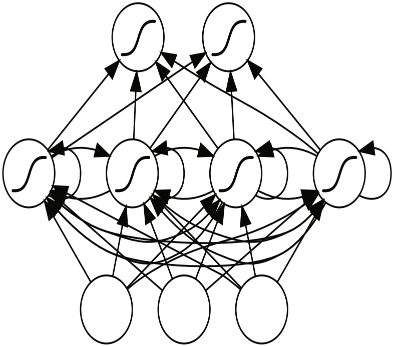


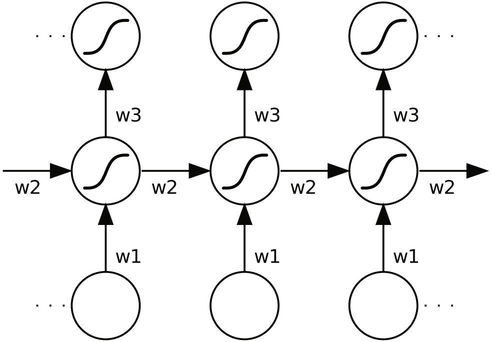


RNN

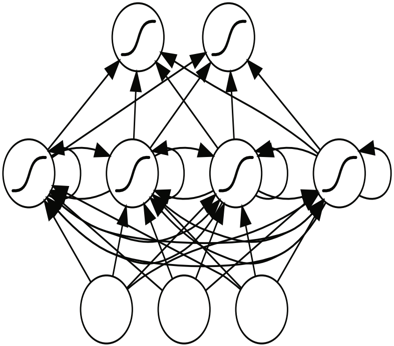


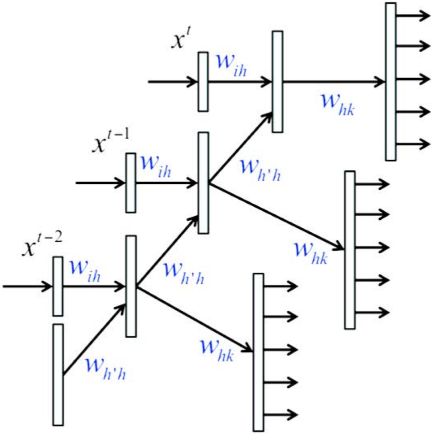
RNN



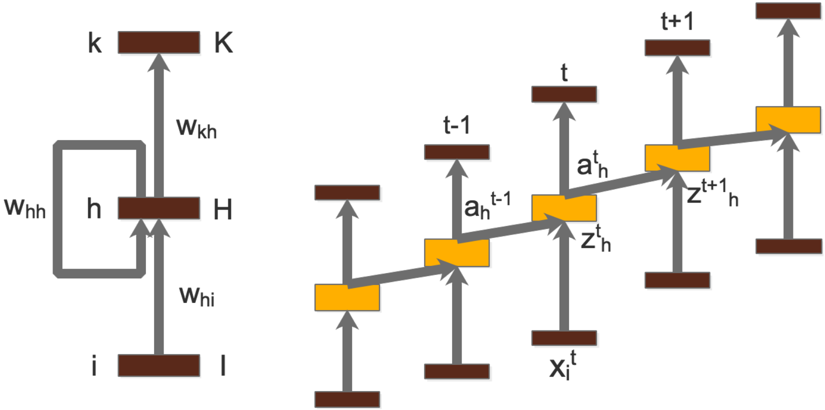


RNN





RNN 符号体系



RNN 前向传播推导

x: 长度为 T 的输入；x_i^t:t 时刻的输入 x 的第 i 维；
I: 输入层神经元个数；H: 隐藏层神经元个数；K: 输出层神经元个数；
z_j^t: 神经元 j 在 t 时刻的待激活输入；
a_j^t: 神经元 j 在 t 时刻的激活值；
J^t: 用 t 时刻的输出计算的代价函数；

$$z_h^t = \sum_{i=1}^{I} w_{hi} x_i^t + \sum_{h'=1}^{H} w_{hh'} a_{h'}^{t-1} \qquad a_h^t = f_h(z_h^t)$$

$$z_k^t = \sum_{h=1}^{H} w_{kh} a_h^t$$

$$J = \sum_{t=1}^{T} J^t(W, b)$$

其中，a_i^0 需要进行初始化，可以选择 0，也可以选择非零初始值。

RNN 后向传播推导

$$w_{kh} = w_{kh} - \frac{\partial J}{\partial w_{kh}}$$

$$= w_{kh} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial w_{kh}}$$

$$= w_{kh} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_k^t} \frac{\partial z_k^t}{\partial w_{kh}}$$

因为：$z_k^t = \sum_{h=1}^{H} w_{kh} a_h^t$ 所以：

$$w_{kh} = w_{kh} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_k^t} a_h^t$$

RNN 后向传播推导

$$w_{hh'} = w_{hh'} - \frac{\partial J}{\partial w_{hh'}} = w_{hh'} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial w_{hh'}}$$

$$= w_{hh'} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_h^t} \frac{\partial z_h^t}{\partial w_{hh'}}$$

因为：$z_h^t = \sum_{i=1}^{I} w_{hi} x_i^t + \sum_{h'=1}^{H} w_{hh'} a_{h'}^{t-1}$ 所以：

$$= w_{hh'} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_h^t} a_{h'}^{t-1}$$

设：$\delta_h^t = \frac{\partial J^t(W,b)}{\partial z_h^t}$ 则：$w_{hh'} = w_{hh'} - \sum_{t=1}^{T} \delta_h^t a_{h'}^{t-1}$

RNN 后向传播推导

$$w_{hi} = w_{hi} - \frac{\partial J}{\partial w_{hi}} = w_{hi} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial w_{hi}}$$

$$= w_{hi} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_h^t} \frac{\partial z_h^t}{\partial w_{hi}}$$

因为：$z_h^t = \sum_{i=1}^{I} w_{hi} x_i^t + \sum_{h'=1}^{H} w_{hh'} a_{h'}^{t-1}$ 所以：

$$= w_{hi} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_h^t} x_i^t$$

设：$\delta_h^t = \frac{\partial J^t(W,b)}{\partial z_h^t}$ 则：$w_{hi} = w_{hi} - \sum_{t=1}^{T} \delta_h^t x_i^t$

RNN 后向传播推导

因为：

$$z_h^t = \sum_{i=1}^{I} w_{hi} x_i^t + \sum_{h'=1}^{H} w_{hh'} a_{h'}^{t-1}$$

所以：

$$\begin{aligned}
\delta_h^t = \frac{\partial J(W,b)}{\partial z_h^t} &= \sum_{k=1}^{K} \frac{\partial J^t}{\partial z_k^t} \frac{\partial z_k^t}{\partial z_h^t} + \sum_{h=1}^{H} \frac{\partial J^{t+1}}{\partial z_h^{t+1}} \frac{\partial z_h^{t+1}}{\partial z_h^t} \\
&= \sum_{k=1}^{K} \frac{\partial J^t}{\partial z_k^t} \frac{\partial z_k^t}{\partial a_h^t} \frac{\partial a_h^t}{\partial z_h^t} + \sum_{h=1}^{H} \frac{\partial J^{t+1}}{\partial z_h^{t+1}} \frac{\partial z_h^{t+1}}{\partial a_{h'}^t} \frac{\partial a_{h'}^t}{\partial z_h^t} \\
&= \sum_{k=1}^{K} \delta_k^t w_{kh} f_h'(\cdot) + \sum_{h=1}^{H} \delta_h^{t+1} w_{hh'} f_h'(\cdot) \\
&= \left(\sum_{k=1}^{K} \delta_k^t w_{kh} + \sum_{h=1}^{H} \delta_h^{t+1} w_{hh'} \right) f_h'(\cdot)
\end{aligned}$$

RNN 后向传播推导

$$w_{hk} = w_{hk} - \sum_{t=1}^{T} \frac{\partial J^t(W,b)}{\partial z_k^t} a_h^t$$

$$w_{hi} = w_{hi} - \sum_{t=1}^{T} \delta_h^t x_i^t$$

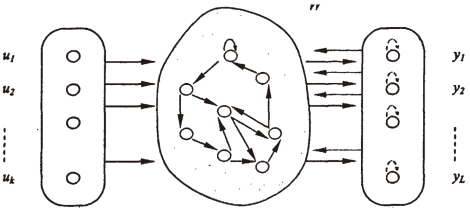
$$w_{h'h} = w_{h'h} - \sum_{t=1}^{T} \delta_h^t a_{h'}^{t-1}$$

$$\delta_h^t = \left(\sum_{k=1}^{K} \delta_k^t w_{kh} + \sum_{h'=1}^{H} \delta_h^{t+1} w_{hh'} \right) f_h'(\cdot)$$

一种常用的RNN网络——ESN

Echo State Networks（ESN）

- Echo State Networks（ESN，回声状态网络）
  - 由Jaeger于2001年提出；
  - 序列模式学习；
  - 对应RNN训练时训练算法复杂，计算量大的问题；



[PDF] The "**echo state**" approach to analysing and training recurrent neural **networks**-with an erratum note

H Jaeger - Bonn, Germany: German National Research ..., 2001 - minds.jacobs-university.de

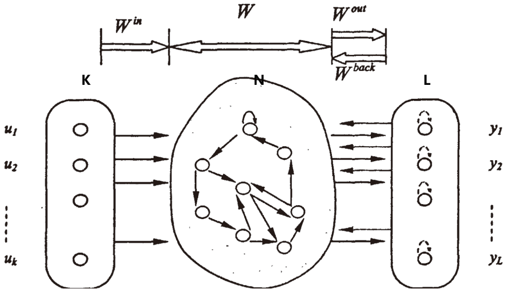
Abstract. The report introduces a constructive learning algorithm for recurrent neural **networks**, which modifies only the weights to output units in order to achieve the learning task. key words: recurrent neural **networks**, supervised learning

Cited by 1058 Related articles All 2 versions Cite Save More

Echo State Networks (ESN)

■ ESN结构

- ◆ 由输入层、中间状态储备池(Dynamic Reservoir, DR)、输出层组成；
- ◆ 状态储备池DR包含了多个随机生成且稀疏连接的神经元；
- ◆ DR使网络具有良好的短期记忆能力和非线性动态特性；

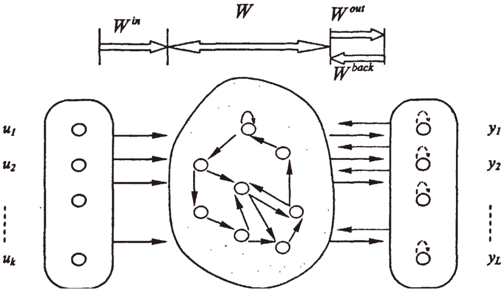


Echo State Networks（ESN）

- W^{in}：N×K维，表示输入层K个神经元与DR的N个神经元的连接；
- W^{out}：(K+N+L)×L维，表示输入层K个神经元、DR的N个神经元分别与输出层L个神经元的连接以及输出层L个神经元的自连接；
- W^{back}：N×L维，表示输出层L个神经元反馈到DR的N个神经元的连接；
- W ：N×N维，表示DR内部的N个神经元的稀疏连接。
  - u —— 输入向量、
  - X —— DR的状态向量
  - y —— 输出向量

$$X(n+1) = f(W^{in}u(n+1) + WX(n) + W^{back}y(n))$$

$$y(n+1) = f^{out}(W^{out}(u(n+1), X(n+1), y(n))$$



Echo State Networks（ESN）

■ 训练过程

- ◆ 训练目标：W^{out}

第一步：初始化

- ◆ 对权值矩阵W^{in}，W^{back}，W 和储备池初始状态$\boldsymbol{x}(0)$进行初始化
  - W^{in}, W^{back}由均匀分布于$[-a, a]$间的随机数进行初始化。
  - $W = \alpha \cdot W_0/|\boldsymbol{\lambda}_{max}|$ 其中，W_0为一个稀疏矩阵，$\boldsymbol{\lambda}_{max}$为$W_0$ 的最大特征值，通常，$\alpha < 1$；
  - 由于网络训练之前，储备池无初始状态，因此可将$\boldsymbol{x}(0)$初始化为0；

Echo State Networks (ESN)

- 训练过程

第二步：更新储备池状态矩阵

- 保持权值矩阵W^{in} , W^{back} , W 不变
- 输入训练样本$(u(1),d(1),\cdots,u(T),d(T))$
- 按照

$$X(n+1)=f(W^{in}u(n+1)+WX(n)+W^{back}d(n))$$

对储备池初始状态$x(n)$进行状态更新；

注意：这里使用的是$(u(1),d(1),\cdots,u(T),d(T))$ 不需要计算$y(n)$

Echo State Networks（ESN）

■ 训练过程

第三步：收集状态向量

- ◆ 选择一个T_0时刻（$T_0 < T$）

【希望T_0时刻的网络状态向量X依赖于训练数据，而不依赖于初始化数据】

- ◆ 收集$T - T_0$时间中的状态向量X，构成一个$(T - T_0 + 1)\times(K + N + L)$维

$$M = [u(T_0), X(T_0), d(T_0); u(T_0+1), X(T_0+1), d(T_0+1); \cdots; u(T), X(T), d(T)]$$

- ◆ 收集$T - T_0$时间中的期望输出矩阵P，经计算，该矩阵为一个$(T - T_0 + 1)\times L$维

$$P = \tanh^{-1}[d(T_0); d(T_0+1); \cdots; d(T)]$$

Echo State Networks（ESN）

■ 训练过程

第四步：权值计算

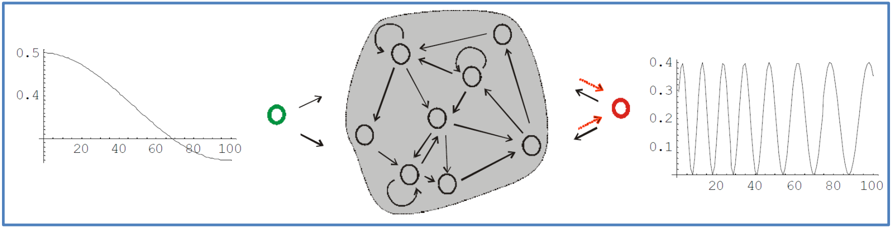
- 由于状态向量和系统输出间呈线性关系，
- 而且，训练的目标是使系统输出$y(k)$尽可能地逼近期望输出$d(k)$，即：

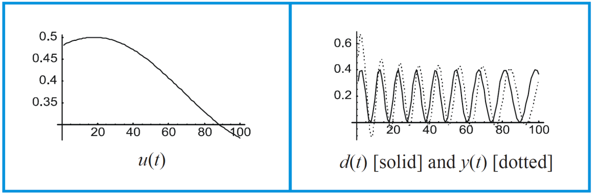
$$d(k) \approx y(k) = \sum_{i=1}^{L} w_i^{out} x_i(n)$$

- 因此，W^{out} 可以通过如下方式求出：

$$(W^{out})^T = M^{-1}P$$

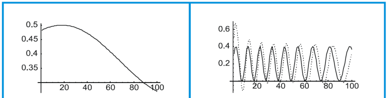
Echo State Networks (ESN)



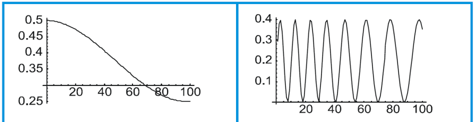


Echo State Networks (ESN)

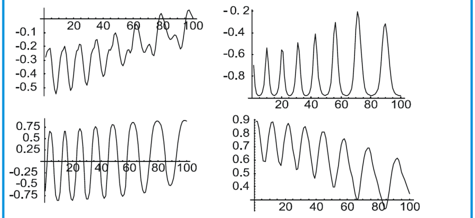
- Traces of some internal DR units.



$u(t)$ $d(t)$ [solid] and $y(t)$ [dotted]



$u(t)$ $d(t)$



four internal states

Echo State Networks（ESN）

- 关于储备池规模N
  - 储备池规模越大，ESN 能表现出的动态特性越复杂，N 的大小直接影响着ESN 的**泛化能力**。
  - N值的选定与被预测数据的信噪比有关
    - 数据信噪比较高时，储备池规模越大，对期望信号的拟合能力也越强。
    - 数据信噪比较低时，储备池规模越大，对预测数据越敏感，网络容易产生“过拟合”现象，会降低模型对测试数据的泛化能力。
- 通常的做法
  - 逐渐增加储备池规模，当网络的处理能力变坏时，即为储备池规模的上限。

Thanks.